Subword Segmental Neural Language Generation for Nguni Languages

FRANCOIS ROLIHLAHLA MEYER

Thesis submitted for the degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

UNIVERSITY OF CAPE TOWN

Supervised by DR JAN BUYS

October 2024

Aan Frans en Maretha

Acknowledgements

I would like to thank my supervisor, Jan Buys, for his mentorship over the past four years. He allowed me to explore the ideas in this thesis as I found interesting, while always guiding me in the right directions. His knowledge and advice proved immensely valuable, especially at times of uncertainty in my research. My PhD journey has been an overwhelmingly positive and rewarding experience, and this has a lot to do with having Jan as my supervisor.

I am grateful to the UCT NLP group for the weekly research discussions and lunches. Our many stimulating conversations certainly informed and improved my research. Thanks to Sello Ralathe, Shane Acton, Neil Sinclair, Maxwell Mojapelo, Yassin Nurmahomed, Nomonde Khalo, Claytone Sikasote, Natalie Alexander, and Zola Mahlaza. Special thanks to Khalid N. Elmadani for being my labmate, collaborator, and conference buddy.

I gratefully acknowledge the Hasso Plattner Institute for Digital Engineering, through the HPI Research School at the University of Cape Town, which funded my PhD studies. I am grateful to all the organisations that provided computational resources for my project. Most of the computations were performed using facilities provided by the University of Cape Town's ICTS High Performance Computing team. I also acknowledge the Centre for High Performance Computing (CHPC), South Africa, for providing compute. Additionally, my research was supported by a grant from the Google Cloud Research Credits program.

During my PhD, I was fortunate to do an internship at the National Institute of Information and Communications Technology (NICT) in Kyoto. Thanks to Raj Dabre for hosting me and for everyone at NICT for making me feel welcome, especially Tetsuko Yamaji. Thanks to my collaborators, Haiyue Song, Abhisek Chakrabarty, and Hideki Tanaka. I thoroughly enjoyed my time in Japan and returned to Cape Town with renewed enthusiasm for my PhD research.

My family and friends have all helped me across the PhD finish line in different ways. I am grateful to you all for supporting me and for patiently nodding whenever I explained my PhD topic. I would like to single out my Observatory friends – thanks to Felix, Erin, Jana, and Martin for accompanying me on my PhD journey from day one. I especially appreciated our after-work meetups during the more demanding periods of my PhD.

I would like to sincerely thank Jürgen Klopp and the Liverpool FC teams of 2021–2024 for being a continued source of inspiration and entertainment. A special mention is also reserved for the wonderful people of The Scrumpy Jack in Observatory.

I am deeply grateful to my parents for supporting my studies and for always being there when I needed them. My mom is a never-ending source of encouragement and lunches, which often gave me the motivation I needed when conference deadlines were ahead. Dankie Maretha. From an early age, my dad sparked my interest in science and nurtured my curiousity about the natural and mathematical world. This set me on a path to pursuing a PhD and a career in research. So far, this journey has been incredibly meaningful and fulfilling. Dankie Frans.

Finally, I want to thank my soon-to-be wife, Igna, for her constant support, her words of encouragement, and for celebrating every PhD milestone with me.

Declaration

I, Francois Rolihlahla Meyer, hereby declare that the work on which this thesis is based is my original work (except where acknowledgements indicate otherwise) and that neither the whole work nor any part of it has been, is being, or is to be submitted for another degree in this or any other university. I authorise the University to reproduce for the purpose of research either the whole or any portion of the contents in any manner whatsoever.

Signature: Signed by candidate Date: 9 October 2024

Abstract

Deep learning models for text generation are now able to produce fluent and coherent text in many conversational settings. However, such models require large training datasets and are primarily designed for a limited number of high-resource languages. These advances are not directly applicable to low-resource languages with distinctive linguistic characteristics. In this thesis we develop text generation models for the Nguni languages of South Africa – isiXhosa, isiZulu, isiNdebele, and Siswati. The Nguni languages are agglutinative and conjunctively written, so words are formed by stringing together morphemes. We design neural models that suit the morphological complexity of the Nguni languages by explicitly modelling the segmentation of words into subword units.

We propose subword segmental modelling, a neural architecture and training algorithm that learns subword segmentation during training. The standard approach to subword modelling is to apply data-driven algorithms such as byte-pair encoding (BPE) during preprocessing. Subword segmental modelling represents a departure from this paradigm: instead of casting subword segmentation as a preprocessing step, we incorporate it into end-to-end learning to allow the model to discover the optimal subword units for a particular language and task. Explicitly modelling the complex subword structure of Nguni languages serves as an inductive bias for more efficient training on the typically limited training data.

In this thesis we present subword segmental models for three natural language generation tasks. Our first model is for autoregressive language modelling. We propose the subword segmental language model (SSLM), a decoder-only model that learns subword segmentation to optimise its language modelling objective. SSLM achieves lower (better) perplexity-based intrinsic evaluation scores than tokenisation-based language models, on average across the four Nguni languages. We also evaluate SSLM as an unsupervised morphological segmenter, showing that its learned subwords are closer to morphemes than standard subword tokens. Since SSLM is our first instantiation of subword segmental modelling, we present a detailed analysis of the architectural components and hyperparameters we found to be influential during development.

Our second model extends subword segmental modelling to neural machine translation (NMT). We propose subword segmental machine translation (SSMT), an encoder-decoder model that learns target language subword segmentation to optimise its sequence-to-sequence translation objective. To generate translations with SSMT, we propose dynamic decoding,

a decoding algorithm for generating text with subword segmental architectures. SSMT outperforms tokenisation-based NMT on Nguni languages, achieving large gains in the extremely low-resource setting of English to Siswati translation. As for SSLM, we show that SSMT learns subword boundaries more aligned with morpheme boundaries than tokenisation-based subwords. SSMT also exhibits greater morphological compositional generalisation, the ability to generalise to novel combinations of known morphemes.

We extend SSMT to multilingual translation, where it learns a single target-side subword segmentation scheme to optimise performance across multiple translation directions. We compare multilingual SSMT to multilingual tokenisation-based NMT. Multilingual SSMT does induce cross-lingual transfer, but to a lesser extent that multilingual tokenisation. In cross-lingual finetuning experiments, SSMT improves transfer between unrelated languages. Our experiments confirm that decisions around subword segmentation greatly affect cross-lingual performance. We also show that differences in orthographic word boundary alignment between languages can impede cross-lingual transfer.

Our third and final model combines subword segmental modelling with a copy mechanism, for the task of data-to-text generation. We propose the subword segmental pointer generator (SSPG), which jointly learns to segment words and copy subwords to optimise data-to-text generation. We also propose unmixed decoding, a text generation algorithm for copy-equipped subword segmental models. On isiXhosa data-to-text, SSPG outperforms tokenisation-based architectures trained from scratch. Besides reference-based evaluation, we develop an extractive evaluation framework to measure how faithfully models capture the expected data content of generations. This shows that SSPG more effectively combines entity copying and morphological composition.

Across all three tasks, and for all four Nguni languages, subword segmental modelling consistently equals or outperforms equivalent tokenisation-based models. Its performance gains are greatest for extremely low-resource languages and tasks. Through linguistically informed evaluations, we show that subword segmental modelling successfully acquires particular aspects of Nguni-language morphology. Its subword units resemble morphemes more closely than subword tokens and it effectively applies morphological composition. Subword segmental modelling proves effective for the Nguni languages, offering a promising new approach to text generation for low-resource, morphologically complex languages.

Contents

Nomenclature

Acronyms / Abbreviations

BPC	Bits-per-character
BPE	Byte-pair encoding
chrF	Character n-gram F-score
DPE	Dynamic programming encoding
LLM	Large language model
LM	Language model
MBI	Morpheme boundary identification
MDL	Minimum description length
MI	Morpheme identification
MT	Machine translation
NCHLT	National Center for Human Technology
NLG	Natural language generation
NMT	Neural machine translation
NT	Neural templates

OBPE Overlap BPE

PG Pointer generator

PLM Pretrained language model

SADiLaR South African Centre for Digital Language Resources

SLM Segmental language model

SSD Subword segmental decoder

SSLM Subword segmental language model

SSMT Subword segmental machine translation

SSPG Subword segmental pointer generator

SSWM Subword segmental word model

T2X Triples-to-isiXhosa dataset

ULM Unigram language model

UMS Unsupervised morphological segmentation

WMT Workshop on Machine Translation

XBPE Extend BPE

1

Introduction

Can grammatical and morphological "rules" be acquired by bottom-up processes, competent and uncomprehending?

Daniel Dennett, *From Bacteria to Bach and Back*, 2017

A long standing goal of AI is to create machines that can generate natural language. This is made most clear by the enduring relevance of the Turing test [152], which framed the quest for creating thinking machines as the quest for creating machines that produce human-like language. Developing such a machine is challenging on many fronts, since humans unconsciously combine multiple cognitive capabilities during language generation. Deciding *what to say* (the meaning of an utterance) requires high-level abstract reasoning. Determining *how to say* something (its surface realisation) requires mastery of a language grammar, as well as substantial planning to construct sentences coherently and fluently.

Until recently, sophisticated language generation was beyond AI, but advances in deep learning for natural language processing (NLP) have enabled unprecedented progress [126, 55]. In large language models (LLMs) [105, 150], we now have systems that produce high-quality text in varied conversational settings, often generating language that is indistinguishable from human-written text [64, 97]. These advances are the latest in a line of research tracing back to neural sequence-to-sequence models [144] only a decade ago, when language generation was limited to more structured tasks like machine translation (MT).

The story of the deep learning NLP revolution, recounted as a series of breakthroughs from sequence-to-sequence translation [144] to GPT-4 [105], paints only part of the full picture. These breakthroughs were made possible by neural sequence models trained on increasingly large, web-scraped corpora. Such datasets are not available for most of the world's 7000+ languages. As a result, many of these developments are only applicable to a small selection of so-called *high-resource* languages. *Low-resource* languages has become an umbrella term for languages that lack large-scale training corpora and diverse evaluation sets. Broadening the reach of deep learning to low-resource languages remains a significant challenge. There is a growing body of research addressing this disparity. Most notably, multilingual modelling [23, 169, 148] extends the advantages of deep learning to a broader range of languages. However, the languages benefiting from this expanded coverage are still limited by data availability. Multilingual modelling has mainly benefited *medium-resource* languages, while truly low-resource languages continue to lag behind [104, 3].

Given the rapid progress achieved for many languages, NLP is now dominated by deep learning, which is inherently data-dependent. For text generation, advances have relied especially heavily on dramatic increases in training sizes [69]. The NLP research landscape has shifted from linguistically motivated modelling to data-driven modelling, wherein language-agnostic learning systems are indiscriminately applied across languages. This has resulted in missed opportunities for improving NLP performance in low-resource languages. The neural techniques that currently pervade NLP were initially developed mainly for English [94]. Their architectures and training regimes encode certain assumptions which, although sufficient for English, are not guaranteed to generalise across topologically diverse languages [11]. Languages vary along many dimensions, some of which are relevant to the design decisions of NLP systems. Such linguistic variation is particularly problematic for low-resource languages, where data scarcity makes accurate modelling assumptions critical.

In this thesis, we address these challenges for one group of languages, the Nguni languages of South Africa – isiXhosa, isiZulu, isiNdebele, and Siswati. We focus on text generation, a category of NLP applications that rely on automatic natural language generation (NLG), such as machine translation, text summarisation, and question answering.

State-of-the-art Nguni text generation systems still exhibit limited capabilities. We design novel neural architectures that suit the linguistic properties of Nguni languages, specifically their complex *morphological* systems. The morphology of a language refers to the processes through which morphemes, the smallest subword units of meaning in a language, are combined to form words. The Nguni languages are *morphologically complex* – their words tend to consist of many morphemes that are concatenated together. NLP systems for the Nguni languages benefit from *subword segmentation* (segmenting words into smaller, subword units for modelling), as it enables generalisation to new words consisting of known subwords.

We introduce *subword segmental modelling*, a neural sequence modelling technique that explicitly models the subword segmentation of a language, for autoregressive or conditional text generation. The core idea of subword segmental modelling is to learn how to segment words during training, allowing the model to discover optimal subword units for a particular language and task. We posit that this would improve deep learning models for the Nguni languages. Their fundamental units of meaning are subwords (morphemes), so finding suitable subword units is crucial for effectively modelling syntax and semantics. Subword segmental modelling would allow models to *learn* which subword units optimise performance.

Subword segmental modelling is a general purpose technique that can be adapted for any neural architecture and sequence modelling task. Broadly, it refers to a combination of three novel modelling techniques:

- **Architectural innovations:** augmenting neural sequence models for text generation with sub-networks that can assign probabilities to arbitrary subword segmentations.
- **Training algorithm:** a dynamic programming algorithm for efficiently incorporating information from multiple subword segmentations during training, enabling the model to learn which segmentations optimise performance.
- **Decoding algorithms:** text generation algorithms that leverage the knowledge encoded in the sub-networks and/or dynamic program of subword segmental models.

In this thesis we present subword segmental models for three Nguni-language tasks: language modelling, machine translation, and data-to-text.

1.1 Motivation

We propose subword segmental modelling as a text generation architecture aimed at modelling low-resource, morphologically complex languages. Here we motivate our choice of focusing on the Nguni languages (the practical application of thesis), and explain how subword segmental modelling (the overarching technical contribution of this thesis) is designed to handle their distinct linguistic properties and data availability.

1.1.1 Application: Nguni Text Generation

The Nguni languages are a related group of Southern African languages with a combined 23.4 million L1 speakers [33], comprising over 43% of the population of South Africa. Developing effective neural text generation models for the Nguni languages is challenging for two reasons. Firstly, the Nguni languages are under-resourced in terms of publicly available datasets for NLP. While isiXhosa and isiZulu have increasingly more datasets available, isiNdebele and Siswati remain extremely low-resourced. None of the languages have the scale of data available for training high-quality, domain-general text generation models like those that now exist for high-resource languages. Secondly, the morphological system of the Nguni languages is *agglutinative* [146]. Agglutinative languages have a high morpheme-to-word ratio, each morpheme encodes a single grammatical role (e.g. a particular case, tense, or singular/plural), and words are formed by concatenating morphemes. The Nguni languages are also written conjunctively, so morphemes are not space-separated but written together as a single orthographic word. As a result, the Nguni languages are characterised by long word forms, consisting of several morphemes which have been strung together. For example, the three-word sentence “I am grateful” translates to the one-word sentence “Ndiyabulela” in isiXhosa, where “Ndi” means “I”, “ya” means “am”, and “bulela” means “grateful”.

The primary linguistic units of meaning in these languages are morphemes – not words. Any attempt to develop NLP systems for the Nguni languages has to be built around the assumption that subwords are the fundamental modelling units. The question then arises: how do we decide which subword units to use? Equivalently, how should words be segmented

into subwords for modelling purposes? This is the question of subword segmentation – the task of transforming a sequence of words into a sequence of subwords. The sentence "I am grateful" can be represented as a sequence of words, a sequence of characters, or as something in between – a sequence of subwords.

```
Words:  I am grateful
Characters:  _I _a m _g r a t e f u l
Subwords option 1:  _I _a m _gra te ful
Subwords option 2:  _I _a m _g rateful
Subwords option 3:  _I _am _grate ful
          ⋮
```

There are many more ways of segmenting this sentence into subwords, depending on where one chooses to insert subword boundaries.[1] The choice represents an important modelling decision – the resulting subword units will be the vocabulary our model is equipped with as it learns the syntactic rules and semantic lexicon of a language. The issue is especially important for agglutinative languages, where linguistic competence depends on being able to decompose its words into meaningful subword units, so the subword segmentation strategy is a critical component of modelling such languages.

The importance of subword segmentation for the Nguni languages is compounded by the fact that they are low-resource languages. A central motivation in moving from word-based modelling to subword-based modelling is to effectively model rare words, which can be composed as a sequence of common subwords. The phenomena of encountering rare or new words after training is unavoidable for low-resource languages. Available datasets are small, so any held-out dataset will contain rare or previously unseen words. Therefore it is critical for models to encode syntactic and semantic information in its subword features and effectively model morphological composition (how the meaning of a word is composed from the meaning of its constituent morphemes).

[1]These examples employ the common practice of appending special tokens (an underscore in this case) to the first subword token from each word. This represents the original word boundaries (spaces) and ensures that the original word sequence can be recovered from any particular subword segmentation.

We argue that the combined effect of data scarcity and morphological complexity has played a significant role in limiting the capabilities of NLP models for the Nguni languages. However, they also highlight a direction of research that we believe could lead to modelling improvements for these languages – reconsidering the task of subword segmentation.

Morphological classification and terminology

The Sotho-Tswana languages of South Africa (Sesotho, Setswana, Sepedi) are also agglutinative, but are written disjunctively – morphemes are separated by spaces and a single linguistic word (e.g. a noun or a verb) may be written as several orthographic words. Therefore, while the morphological systems of both the Nguni and Sotho-Tswana language are classified as agglutinative, their orthographies (writing systems) insert word boundaries at very different granularities of linguistic meaning. The Nguni languages produce long written words, while the Sotho-Tswana languages have visibly shorter words in comparison. We return to this difference between conjunctive and disjunctive orthographies in Chapter 5.

Throughout this thesis we refer to the languages we are targeting with our methods as "agglutinative". Unless otherwise specified, this excludes agglutinative languages with disjunctive orthographies, like the Sotho-Tswana languages, which represent an unusual mismatch between linguistic and orthographic word boundaries. Our techniques are geared towards languages characterised by many morphemes per written word, since that is where subword segmentation becomes crucial.

We will sometimes use the term "morphologically complex", instead of "agglutinative", to refer to the languages we are targeting. By this we simply mean languages with high morpheme-to-word ratios. Our primary target is agglutinative languages with conjunctive orthographies, which is a subset of morphologically complex languages. For example, *fusional* languages are also morphologically complex (high morpheme-to-word ratio) but morphemes are fused together (they undergo surface-level changes), instead of being concatenated as they are in agglutinative languages.

1.1.2 Contribution: Subword Segmental Modelling

To explain our reasoning in proposing subword segmental modelling as an answer to the modelling challenges posed by the Nguni languages, we first summarise the standard approach to subword modelling in NLP, namely subword tokenisation, and discuss its drawbacks. Subword tokenisation has been widely adopted in NLP. Several algorithms have been proposed, with byte-pair encoding (BPE) [135] and the unigram language model (ULM) [81] among the most popular. BPE builds a vocabulary based on training corpus subword frequency, and segments words into frequently occurring subwords. ULM builds a vocabulary based on the training corpus likelihood under a unigram language model, and segments words into subwords that maximise this likelihood. Both of these algorithms exemplify the dominant paradigm in NLP: subword segmentation is cast a preprocessing step in which text is tokenised into subwords.[2] Tokenisers are applied to datasets before models are trained on the segmented text.

There are downsides to relegating subword segmentation to the domain of preprocessing. The algorithms are task-agnostic. BPE is essentially a compression algorithm [44], while the probabilistic model underlying ULM assumes that subwords are context-independent, which is not a suitable inductive bias for NLP modelling. None of these subword methods consider the NLP task for which the subword units will eventually be used e.g. language modelling or MT. Instead, they learn subwords that optimise some corpus-based statistical criterion such as co-occurrence frequency (in the case of BPE) or unigram-based likelihood (in the case of ULM). These criteria encode assumptions that might aid modelling to some extent, such as capturing frequent subwords that are likely to be morphemes, but they do not guarantee optimal subword segmentation for task performance.

The segmenters are also language-agnostic. While language-specific pre-tokenisers are sometimes employed for sentence tokenisation [77], subword tokenisers are applied without change to different languages, even though different morphologies might require different

[2]The terms "subword segmentation" and "subword tokenisation" is used interchangeably in the literature, with the latter being more common. We will refer to "subword segmentation" as the general task of modelling language with subword units, and "subword tokenisation" as the more specific strategy of segmenting words into subwords during preprocessing, which is the standard approach in NLP. We will also refer to algorithms like BPE and ULM as "subword tokenisers".

segmentation techniques. One option would be to apply morphological segmenters, trained or rule-based, as subword tokenisers. However, accurate morphological segmenters are not readily available for low-resource languages. Furthermore, morphological segmentation can produce prohibitively large vocabularies since long morphemes, such as root words and named entities, are left unsegmented. In practice, morphological boundaries are sometimes incorporated into existing algorithms like BPE [10, 60], which fails to address the more fundamental problems of subword tokenisation.

These issues arise from the fact that the subword tokenisation operates completely independent of model training. Models are reliant on the quality of the subwords produce by tokenisers. If the subwords are suboptimal units for modelling, the neural networks cannot change the subword boundaries – they are fixed after preprocessing. Ideally subword segmentation should be part of the learnable parameters of a model, so that it can be adjusted to optimise the training objective. This would allow neural networks to *learn* the optimal subword segmentation strategy for a specific language and task, and adjust subword segmentation to maximise performance. By shifting subword segmentation from preprocessing to training, the model would be able to "discover" the optimal segments for sequence prediction.

1.2 Aims & Hypotheses

The overarching goal of this thesis is to improve the Nguni-language text generation capabilities of deep learning models. Our approach is to develop new neural network architectures that target a specific linguistic property of the Nguni languages – their agglutinating morphological system. The technique we propose to achieve this is subword segmental modelling, a new class of neural network architectures and training algorithms that explicitly model and optimise subword segmentation during training. To establish subword segmental modelling as a neural architecture, we define the aims of this thesis as follows:

- Develop subword segmental neural architectures that jointly model subword segmentation alongside autoregressive language modelling and sequence-to-sequence modelling.
- Develop decoding algorithms for generating natural language with trained subword segmental models.
- Analyse the performance and linguistic knowledge gained by subword segmental models when applied to Nguni text generation tasks across languages and data sizes.

Our thesis is that learning subword segmentation during training, instead of fixing it after preprocessing as in subword tokenisation, will prove beneficial for modelling low-resource, morphologically complex languages. Subword segmental modelling adjusts the subword segmentation scheme of a model to optimise performance, and it stands to reason that this holds particular promise for languages with complex subword structures. We turn to the Nguni languages to test these ideas, putting forth the following hypotheses:

- Subword segmental modelling will improve performance on Nguni text generation tasks by more effectively modelling the complex subword structure of Nguni languages.
- Explicitly modelling subword segmentation is a useful inductive bias for agglutinative languages, which will allow models to leverage limited data more effectively, thereby improve performance on low-resource languages/tasks.
- Subword segmental models will acquire linguistically sound knowledge about the morphological systems of languages, even without explicit morphological supervision.

1.3 Contributions

In summary, the main contributions of this thesis are as follows:

- We propose subword segmental language modelling (SSLM) to unify autoregressive language modelling and subword segmentation, thereby improving perplexity-based evaluation metrics for Nguni language modelling.

- We propose subword segmental machine translation (SSMT) to unify machine translation and subword segmentation, thereby improving translation from English to Nguni languages and leading to large gains in the extremely low-resource of English to Siswati translation.

- To perform inference with SSMT we propose dynamic decoding, a text generation algorithm that adapts segmentations as it generates translations.

- We show that SSLM and SSMT act as unsupervised morphological segmenters, learning subwords that resemble morphemes more than tokenisation-based subwords.

- We introduce the task of morphological compositional generalisation to test generalisation to previously unseen combinations of morphemes, and show that SSMT generalises more reliably in this way.

- We present a systematic performance analysis of SSMT and other subword methods in multilingual and cross-lingual translation, showing that SSMT shows no benefit for multilingual modelling but proves useful for cross-lingual finetuning.

- We propose the subword segmental pointer generator (SSPG), a data-to-text model that unifies sequence-to-sequence learning, entity copying, and subword segmentation, thereby improving data-to-text generation for agglutinative languages over tokenisation-based models trained from scratch and finetuned PLMs.

- We introduce an extractive evaluation framework for isiXhosa data-to-text, which demonstrates that SSPG copies data entities more reliably than comparable models and is able to append morphemes to copied entities according to the morphological rules of isiXhosa.

- For each of our models we report the role of different hyperparameters and perform ablation studies, which highlight the importance of each of the architectural components that make up subword segmental modelling.

1.4 Thesis Outline

Chapter 2: Background

We survey the existing lines of research that are relevant to this thesis. We start by briefly reviewing LSTMs and Transformers. Next we cover the state of subword segmentation in NLP, reviewing popular subword tokenisers and highlighting their shortcomings. We then introduce segmental sequence modelling, which serves as the technical inspiration for our approach to subword segmentation. Lastly, we survey prior work on neural text generation for the Nguni languages.

Chapter 3: Subword Segmental Language Modelling

We present the subword segmental language model (SSLM), which learns how to segment words while being trained for autoregressive language modelling. We train our model on small, high quality datasets for the four Nguni languages. Based on perplexity-based evaluation, SSLM outperforms tokenisation-based language models on average across the Nguni languages. Furthermore, its learned subwords resemble morphemes more closely than popular subword tokens. We also train our SSLM as a word-level sequence model, resulting in an unsupervised morphological segmenter that outperforms existing methods by large margins. SSLM confirms that subword segmental modelling is practically viable, can improve probabilistic sequence modelling for the Nguni languages, and acquires knowledge about morphological boundaries in the process.

Chapter 4: Subword Segmental Machine Translation

We present subword segmental machine translation (SSMT), which unifies subword segmentation and translation in a single trainable model. We also present *dynamic decoding*, a decoding algorithm that enables text generation with subword segmental models. We train and evaluate SSMT across six translation directions (including English to three Nguni languages) and show that it matches or improves chrF scores for morphologically complex languages. It achieves especially large gains in the very low-resource setting of English to Siswati translation (+2.5 chrF). SSMT also learns subwords that are closer to morphemes

than baseline subword tokens and proves more robust on a test set constructed for evaluating morphological compositional generalisation (the ability to generalise to novel combinations of know morphemes). SSMT shows that subword segmental modelling can be incorporated into sequence-to-sequence modelling, proves effective when used for Nguni-language text generation, and is particularly beneficial for extremely low-resource languages.

Chapter 5: Multilingual Subword Segmental Machine Translation

In this chapter we study how SSMT combines with two other popular approaches to low-resource MT, namely multilingual modelling and cross-lingual adaptation. SSMT fails to outperform tokenisation-based multilingual MT, but does prove competitive for cross-lingual finetuning. It outperforms all our baselines on adapting an MT model from a morphologically simple language (Afrikaans) to Nguni languages. The study conducted in this chapter serves as a more general investigation into the role of subword methods and interacting languages in cross-lingual transfer, revealing useful findings for practitioners. The broader findings confirm that decisions around subword modelling can be key to optimising the benefits of multilingual modelling. Subword regularisation leads to greater cross-lingual transfer in multilingual MT, whereas BPE is better suited for adapting an MT model to new translation directions. Our results also suggest that cross-lingual transfer is impeded when languages have misaligned word boundaries (the morphological granularity of written words).

Chapter 6: Subword Segmental Pointer Generator

We present the subword segmental pointer generator (SSPG), which augments subword segmental modelling with a copy mechanism for the task of data-to-text. We also present a new decoding algorithm for SSPG called unmixed decoding. SSPG outperforms existing dedicated data-to-text models for two morphologically complex languages (isiXhosa and Finnish), improving chrF++ scores by more than 2.0 and BLEU scores by more than 1.0. By developing an extractive evaluation framework for isiXhosa data-to-text, we show that SSPG more effectively acquires the ability to attach prefixal morphemes to copied entities. We also compare SSPG to pretrained solutions for data-to-text, where standard PLMs come up short. SSPG is only outperformed by the unconventional approach of fine-tuning MT

models. SSPG demonstrates the feasibility of combining subword segmental modelling with additional inductive biases to enable the acquisition of task-specific morphological modelling.

Chapter 7: Conclusion

We conclude by summarising our contributions and contextualising them within the broader NLP research landscape. First we review the key results from each chapter, before highlighting the common themes that have emerged in our findings about subword segmental modelling across different tasks and languages. Next we propose several potential directions of future research, including suggestions for extending and improving subword segmental modelling. Finally, we end with a discussion of how the work presented in this thesis aligns with current trends in NLP research.

1.5 Publications

The core research presented in this thesis was originally published in the following papers.

- Francois Meyer and Jan Buys. 2022. Subword Segmental Language Modelling for Nguni Languages. In Findings of the Association for Computational Linguistics: EMNLP 2022, pages 6636–6649, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

  Chapter 3 is based on this research.

- Francois Meyer and Jan Buys. 2023. Subword Segmental Machine Translation: Unifying Segmentation and Target Sentence Generation. In Findings of the Association for Computational Linguistics: ACL 2023, pages 2795–2809, Toronto, Canada. Association for Computational Linguistics.

  Chapter 4 is based on this research.

- Francois Meyer and Jan Buys. 2024. Triples-to-isiXhosa (T2X): Addressing the Challenges of Low-Resource Agglutinative Data-to-Text Generation. In Proceedings

of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pages 16841–16854, Torino, Italia. ELRA and ICCL.

Chapter 6 is based on this research.

- Francois Meyer and Jan Buys. 2024. A Systematic Analysis of Subwords and Cross-Lingual Transfer in Multilingual Translation. In Findings of the Association for Computational Linguistics: NAACL 2024, pages 2194–2200, Mexico City, Mexico. Association for Computational Linguistics.

  Chapter 5 is based on this research.

During the course of my PhD, I also participated in collaborations resulting in the following publications. These papers are not central to the aims of this thesis, but they influenced my PhD research and in some cases provided additional baseline results to compare our models against.

- Franziska Pannach, Francois Meyer, Edgar Jembere, Dlamini, Sibonelo Zamokuhle. 2021. NLAPOST2021 1st Shared Task on Part-of-Speech Tagging for Nguni Languages. In Proceedings of the International Conference of the Digital Humanities Association of Southern Africa (DHASA) 2021. Virtual conference with shared task jointly organised by DHASA and SACAIR.

- Khalid Elmadani, Francois Meyer, and Jan Buys. 2022. University of Cape Town's WMT22 System: Multilingual Machine Translation for Southern African Languages. In Proceedings of the Seventh Conference on Machine Translation (WMT), pages 1039–1048, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

- Francois Meyer, Haiyue Song, Abhisek Chakrabarty, Jan Buys, Raj Dabre, and Hideki Tanaka. 2024. NGLUEni: Benchmarking and Adapting Pretrained Language Models for Nguni Languages. In Proceedings of the 2024 Joint International Conference on

Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pages 12247–12258, Torino, Italia. ELRA and ICCL.

The models produced by this research are used as baselines in Chapter 6.

- Haiyue Song, Francois Meyer, Raj Dabre, Hideki Tanaka, Chenhui Chu, Sadao Kurohashi. SubMerge: Merging Equivalent Subword Tokenizations for Subword Regularized Models in Neural Machine Translation. In Proceedings of the 25th Annual Conference of the European Association for Machine Translation (EAMT), pages 147–163, Sheffield, United Kingdom. European Association for Machine Translation.

2

Background

This chapter provides an overview of the previous research on which this thesis builds, introducing the reader to concepts and ideas that are central to our work. Firstly, we briefly cover the established neural networks for NLP, namely LSTMs and Transformers, focusing on the components of their architectures that we repurpose for our subword segmental architectures. Next, we introduce the now universal approach of subword segmentation, surveying a few popular tokenisation algorithms. We then summarise the technical details behind segmental sequence modelling – the general sequence modelling technique which we adapt for subword modelling in this thesis. Lastly, we survey efforts to develop deep learning models for low-resource languages, highlighting existing work involving Nguni languages.

2.1 Neural Networks for NLP

2.1.1 LSTM

Long short-term memory networks (LSTMs) [58] are a type of recurrent neural network (RNN). They process sequences one unit at a time, updating an internal *hidden state* (a vector that encodes the sequence up to its current position). The hidden state is updated by applying *gates* to the hidden state of the previous step, and to the vector representation of the current input. The gates are operations that essentially filter information, determining what

is discarded (forget gate), what is retained (input gate), and what is incorporated into the current hidden state (output gate). At step t in a sequence, the hidden state $\mathbf{h}_t$ is computed as

$$\begin{aligned}\tilde{\mathbf{c}}_t &= \tanh(W_c\mathbf{x}_t + U_c\mathbf{h}_{t-1} + \mathbf{b}_c),\\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t,\\ \mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t),\end{aligned}$$

where $\mathbf{x}_t$ is the input embedding for the t^{th} token, $\mathbf{c}_t$ is the current *cell state*, and $\mathbf{f}_t, \mathbf{i}_t, \mathbf{o}_t$ are the forget, input, and output gates respectively, computed as

$$\begin{aligned}\mathbf{f}_t &= \sigma(W_f\mathbf{x}_t + U_f\mathbf{h}_{t-1} + \mathbf{b}_f),\\ \mathbf{i}_t &= \sigma(W_i\mathbf{x}_t + U_i\mathbf{h}_{t-1} + \mathbf{b}_i),\\ \mathbf{o}_t &= \sigma(W_o\mathbf{x}_t + U_o\mathbf{h}_{t-1} + \mathbf{b}_o).\end{aligned}$$

The weight matrices W, U and bias vectors $\mathbf{b}$ are the trainable parameters of an LSTM. The hidden state $\mathbf{h}_t$ encodes the sequence up to the t^{th} step, and can be passed to further neural layers that produce task-specific output (e.g. predicting the class of the sequence, or the next value in the sequence).

In the case of text generation, an LSTM is trained on text sequences such as sentences represented as sequences of words. During training, the model is repeatedly tasked with predicting the next word in a sequence, requiring it to extract information about the preceding context that informs its prediction of the next word. Through this procedure, the LSTM learns to encode information in the hidden state that tracks long range dependencies in natural language. For example, if a trained LSTM has to predict the next word in the sentences "I am in Paris, so everyone is speaking..." it might successfully predict "French" since its hidden state encodes the fact that Paris was mentioned earlier in the sentence, and during training the weights of the LSTM have internalised that Paris is associated with the French language.

For a few years LSTMs were considered the state-of-the-art for text generation tasks, including machine translation [145] and language modelling [65]. However, it soon became

clear that they had certain limitations which impeded further progress through scaling. Firstly, they process input sequentially and cannot be fully parallelised, leading to excessive training times. Secondly, in practice LSTMs cannot reliably track dependencies that are very far apart in a sequence. The hidden state can become a bottleneck [8], since all the relevant information in a sequence must be encoded in a single fixed length vector. This limits the length and complexity of dependencies that can be modelled by LSTMs.

While LSTMs have been eclipsed by the superior long range capabilities of Transformers [154], they are still preferred in certain scenarios, as exemplified by the following use cases.

- LSTMs outperform Transformers as language models in certain low-resource settings, achieving lower test set perplexity on low-resource South African languages [91] and domain-specific speeh transcriptions [137].
- LSTMs are used as subnetworks for modelling short sequences, such as generating subwords [143, 70, 32, 160]) or short character sequences [86, 37, 36], since in such settings the long range dependency tracking of Transformers is redundant.
- LSTMs are still the architecture of choice for training neural networks from scratch for structured text generation tasks with limited training sets, such as data-to-text generation [165, 166, 138, 121].

2.1.2 Transformer

The key technical innovation of the Transformer architecture [154] is the idea of *self-attention*, whereby the model *attends* to previous tokens in a sequence. Unlike LSTMs, Transformers can directly access the representations of all preceding inputs at any point in a sequence. This is implemented through the *attention mechanism*, which is a dynamically weighted sum of the preceding input representations. During training, a Transformer sequence model learns a weighting scheme that is conditioned on preceding context. This effectively allows the model to learn which preceding tokens are relevant to its task at a specific point in a sequence. In the case of text generation, its task is to predict the next word, so the attention mechanism tracks the syntactic and semantic dependencies of natural language. Returning to

the previous example, if a Transformer has to predict the next word in the phrase "I am in Paris, so everyone is speaking..." it might successfully predict "French" since the attention mechanism at the word "speaking" assigns a high attention weight to the preceding mention of "Paris" (it *attends* to it), and during training it has internalised the association between Paris and the French language.

The attention mechanism computes three vectors for each token in a sequence. These are called the query vector, key vector, and value vector, and are respectively computed as

$$\mathbf{q}_i = W_q \mathbf{x}_i,$$
$$\mathbf{k}_i = W_k \mathbf{x}_i,$$
$$\mathbf{v}_i = W_v \mathbf{x}_i,$$

where $\mathbf{x}_i$ is the input embedding for the i^{th} token and W_q, W_k, W_v are trainable weight matrices. The attention weight α_{ij} of the i^{th} input token and the j^{th} context token is computed as the scaled dot product of $\mathbf{q}_i$ and $\mathbf{k}_j$, passed through a softmax function such that the attention weights sum to 1. The output embedding for the i^{th} token is then computed as a weighted sum of context value vectors,

$$\mathbf{o}_i = \sum_{j=1}^{i} \alpha_{ij} \mathbf{v}_j.$$

These output embeddings $\mathbf{o}_i$ are passed to a feed-forward neural network layer. Transformer models consist of several such self-attention and feed-forward layers stacked on top of each other. The output embeddings produced by the final layer of a Transformer encode information about individual tokens and the sequence as a whole, and can be passed to further neural layers that produce task-specific output.

Transformers overcome the limitations that constrain LSTM scaling. Attention avoids the bottleneck of the LSTM hidden state [8], since directly attending to previous tokens avoids having to encode an entire sequence history in a single recursively updated hidden representation. Since Transformers are parallelisable, they can also be trained more efficiently

than LSTMs. This makes it feasible to train large Transformer models, consisting of multiple layers of attention stacked on top of each other. Such increased depth, in turn, enables the model to track higher level, more complex dependencies. Because of these advantages, Transformers quickly replaced LSTMs as the state-of-the-art deep learning architecture for several NLP tasks.[1]

The most significant impact of Transformers has been in enabling the training of pre-trained language models (PLMs) like BERT [29] and T5 [123], which paved the way for the development of large language models (LLMs) like GPT-4 [105] and LLaMa [150]. These models are all instances of Transformer-based architectures trained on increasingly large web-scraped corpora. This large-scale pretraining has led to the emergence of impressive linguistic capabilities, which enables these models to be adapted to a wide variety of tasks through finetuning [29] or in-context learning [12]. The scaling of both model size and training data enabled by Transformers has been central to these advances.

2.1.3 Encoder-Decoder vs Decoder-Only

LSTMs and Transformers can be used interchangeably as components in more general classes of neural models. The two dominant neural paradigms for text generation are encoder-decoder models and decoder-only models. The choice of architecture depends on the nature of the text generation task.

Decoder-only Decoder-only models consist of a single neural network for sequence modelling. Models are trained to predict the next value in a sequence, given the history of the sequence so far. At each position i in the sequence, the neural network (e.g. LSTM or Transformer) computes a probability distribution over all possible next values,

$$p(y_i|\mathbf{y}_{<i}), \tag{2.1}$$

[1] NLP leaderboards like GLUE (https://gluebenchmark.com/ and Chatbot Arena (https://chat.lmsys.org/), which compare model performance across different tasks, is dominated by Transformer models.

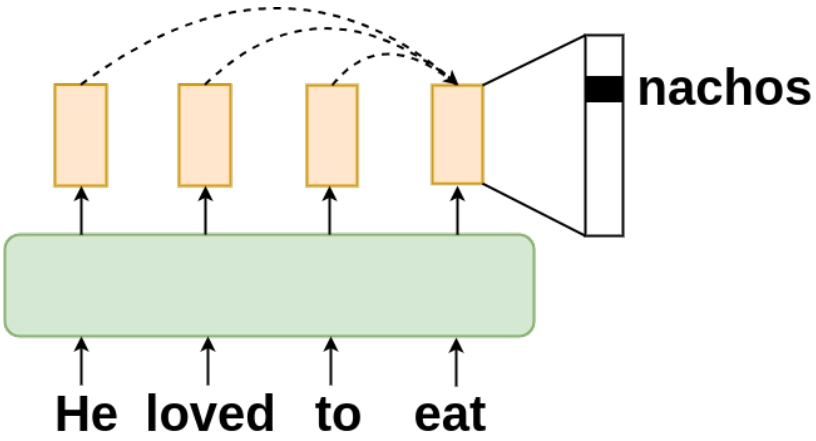

Figure 2.1: The standard decoder-only Transformer architecture used for autoregressive language modelling. The self-attention mechanism encodes the context relevant to next-word generation by attending to tokens already generated.

where the probability is conditioned on the sequence history $\mathbf{y}_{<i}$. In the case of LSTMs this conditioning is achieved by continually passing on the hidden state of the previous step. In the case of Transformers it is achieved by having the decoder attend to tokens in the preceding sequence history (see Figure 2.1), also known as self-attention.

Decoder-only models do not explicitly separate the processing of input and output sequences but instead rely on the sequentially revealed context to generate the next token in a sequence. This setup is used in text generation tasks where the goal is to continually generate the next word in single text sequence, given the preceding words that have been generated so far, such as text continuation [122], dialogue completion [172], and prompt-based generation [108, 105, 150, 63]. Across all of these tasks, the input does not strictly determine or limit the output but rather guides the direction of the expected output text.

Encoder-decoder Encoder-decoder models consist of two subnetworks - an encoder and a decoder. The encoder processes an input sequence $\mathbf{x} = x_1, x_2, ..., x_{|\mathbf{x}|}$, creates an encoding $\mathbf{h}$ of the input which is passed to the decoder, and the decoder generates an output sequence $\mathbf{y} = y_1, y_2, ..., y_{|\mathbf{y}|}$ based on the input. At each position i in the output sequence, the decoder

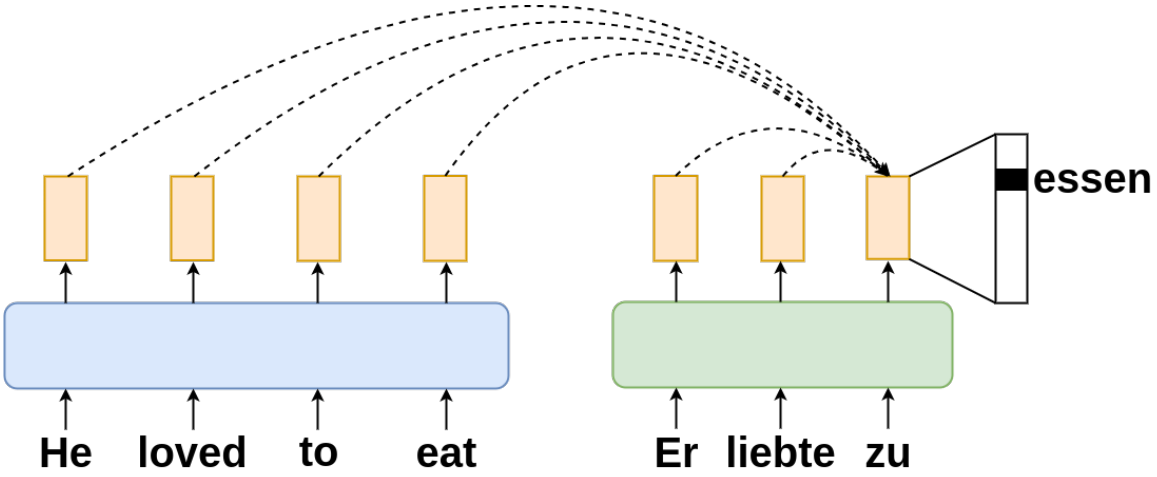

Figure 2.2: The standard encoder-decoder Transformer architecture for sequence-to-sequence tasks like machine translation. The encoder processes an English sentence and the decoder generates a German translation, with dotted lines visualising cross-attention (attending to the English sentence) and self-attention (attending to already generated German words).

computes a probability distribution over all possible next values,

$$p(y_i|\mathbf{y}_{<i}, \mathbf{x}), \tag{2.2}$$

where the conditioning on $\mathbf{y}_{<i}$ is achieved as in decoder-only modelling and conditioning on $\mathbf{x}$ is achieved by passing the encoder representation $\mathbf{h}$ to the decoder network. In the case of LSTMs this is achieved by initialising the decoder LSTM hidden state with the final hidden state of the LSTM encoder. In the case of Transformers this is achieved by having the decoder attend to encoder representations (see Figure 2.2), also known as cross-attention.

This setup is known as sequence-to-sequence modelling – the goal is to map an input sequence $\mathbf{x}$ to an output sequence $\mathbf{y}$, and the two sequences are processed by separate subnetworks. Many text generation tasks can be framed as sequence-to-sequence tasks including machine translation [144] (e.g. mapping English sentences to German translations), text summarisation [127] (e.g. mapping documents to summarising paragraphs), and data-to-text [165] (e.g. mapping flattened data tables to text describing the data). These are all rather structured text generation tasks, where the aim is to learn a constrained mapping function from input text to output text.

2.2 Subword Segmentation

Alongside developments in neural network architectures, the shift from word-based sequence modelling to subword-based sequence modelling has been instrumental in recent NLP advances [95]. Earlier work applying deep learning to NLP processed text as a sequence of words [96] or characters [73]. The choice represents a trade-off. Using words enables the modelling of word-level semantic and syntactic properties, but requires a prohibitively large embedding matrix to represent the vocabulary of all distinct words in the training set. Using characters ensures a feasible vocabulary size, but makes it harder to model higher-level linguistic properties. A middle ground between words and characters is achieved with *subword segmentation*, wherein each word is split into one or more subwords during preprocessing (e.g. "taking it easy" becomes "_ta k ing _it _eas y"). Models are trained on sequences of subword tokens with special characters (e.g. underscores) demarcating the start of new words to make the original space-separated word sequence recoverable during post-processing.

```
Word-based modelling:  taking it easy
Subword-based modelling:  _ta k ing _it _eas y
Character-based modelling:  _t a k i n g _i t _e a s y
```

Subword segmentation leads to smaller vocabularies than word-based modelling, since the number of distinct subwords in a training set will be less than the number of distinct words. But unlike character-based modelling, it does not achieve this at the cost of resorting to entirely meaningless units of text. Subword tokens are sufficiently coarse-grained so as to represent meaningful linguistic units. For example, in subword-based modelling some words will not be segmented at all, with the word itself being represented as a single token (e.g. short words like "it" in the example above), enabling the modelling of some word-level features. Where words are segmented, the resulting subwords will sometimes correspond to morphemes (meaningful subword units e.g. the suffix "–ing" in the example above which indicates present participle form), enabling the modelling of morphological features.

The other major advantage of subword modelling is that it solves the open vocabulary problem. This refers to the challenge of encountering words during evaluation that rarely or never occurred during training, and are therefore not modelled in a fixed word-based vocabulary. With subword modelling, such out-of-vocabulary (OOV) words are represented as a sequence of known subwords, which in some cases would allow the model to compose the meaning of a previously unseen word. For example, if a model is only exposed to the singular form of "dog" during training but has acquired the meaning of the suffix "–s" from other examples of plural nouns in is training set (e.g. "cat–s", "door–s"), it could plausibly compose the meaning of the previously unseen form "dogs".

Subword tokenisation, the approach of applying subword segmentation during preprocessing, has become standard practice in NLP since its application in large pretrained language models like BERT [29] and GPT-3 [12]. The process is guided by data-driven algorithms that construct a subword vocabulary for a training corpus according to some statistical criterion (e.g. subword frequency). This avoids the need for manually designed subword segmentation schemes and creates tokenisers that automatically capture statistical features of the training data. Many subword tokenisation algorithms have been proposed [95], though only a few have gained widespread adoption. We now review the most commonly used tokenisers, discuss the broader class of methods to which they belong, and explore alternative paradigms for subword modelling.

2.2.1 Subword Tokenisers

The most popular subword method in NLP is byte-pair encoding (BPE) [135]. The goal of BPE is to represent common characters sequences (frequent subwords) as distinct vocabulary items. Starting with an initial vocabulary of all unique characters in a training corpus, it iteratively merges items in the vocabulary that co-occur most often, and adds the merged items to the vocabulary. This is repeated until a pre-specified vocabulary size has been reached, at which point BPE can be used as a segmenter. Words are segmented by iteratively merging characters and then subwords (in the same order that the merges occurred during vocabulary construction) until no more merges are posible. BPE is essentially a corpus compression

algorithm [44]. Its merging procedure greedily solves a combinatorial optimisation problem [175], with an objective unrelated to linguistic properties or downstream tasks.

Another popular method is unigram language model (ULM) [81], which aims to maximise the likelihood of a training corpus under a unigram-based language model in which subwords are generated independently. The algorithm starts with a large initial vocabulary, usually consisting of the most frequent corpus substrings, and iteratively removes items to maximise the likelihood of a training corpus under a ULM until the pre-specified vocabulary size has been reached. Usually, words are subsequently segmented by computing the most likely segmentation under the ULM using the Viterbi algorithm [157].

It is also possible to sample subword segmentations from ULM, introducing stochasticity to the segmentation procedure. In this way ULM can be used as a *probabilistic segmenter* (one word can have multiple possible segmentations in the same training corpus), as opposed to a *determinsitic segmenter* like BPE (each word is always segmented identically). The benefit of probabilistic segmentation is that it exposes the model to varying segmentations during training, which reduces sensitivity to particular word segmentations. This is a form of regularisation which improves model robustness with respect to segmentations, varying typographical conventions (e.g. punctuation, spelling), and other types of noise [81].

Besides BPE and ULM, which are both implemented in the widely used open-source SentencePiece library [82], a few other subword methods have gained traction among practitioners. WordPiece [133], which was used for BERT [29], builds a vocabulary by iteratively merging subwords that lead to the greatest likelihood increase under an n-gram language model. BPE-dropout [118] adapts BPE for subword regularisation, introducing stochasticity by randomly dropping some proportion of merges during segmentation. Other subword methods have been proposed for specific applications, such multilingual modelling [112] and fine-tuning [161].

There have also been efforts towards linguistically motivated subword tokenisation. This is achieved by incorporating morphological information into tokenisation algorithms like BPE and ULM to produces subword tokens that are more aligned with morpheme boundaries. This has been successfully applied in MT [60, 106, 131, 130] and language modeling [110, 59, 10].

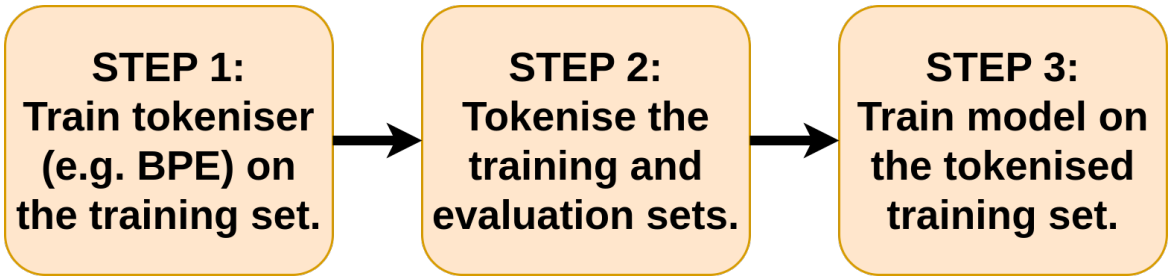

Figure 2.3: The standard NLP modelling pipeline, with subword segmentation applied during preprocessing and finalised before model training.

However, this method depends on the existence of high-quality morphological segmenters or morphologically segmented data for supervised training. Such linguistic resources are not available for most low-resource languages.

An alternative paradigm is unsupervised morphological segmentation, where segmenters are derived from unannotated text corpora by applying information theoretic principles to identify morpheme-like subword segments. For example, language model entropy has been used to estimate morpheme boundaries [99, 100, 98], as entropy tends to increase at the start of a new morpheme [39]. A well-known unsupervised morphological segmenter is Morfessor [25], which maximises a unigram language model likelihood regularised by a minimum description length (MDL) prior that encourages fewer and shorter subwords. Morfessor combines the compression objective of BPE with the language modelling optimisation of ULM, but its MDL prior biases it towards subwords that are more likely to correspond to morphemes.

These methods, as well as the vast majority of recently proposed methods for subword segmentation, all follow the same ubiquitous paradigm of subword tokenisation i.e. *subword segmentation as preprocessing*, wherein subword segmentation is viewed as the first two steps in the modelling pipeline shown in Figure 2.3.

2.2.2 Shortcomings

The widespread adoption of these subword tokenisation algorithms is testament to their effectiveness, but they are not universally applicable. Zhu et al. [174] demonstrate that

existing segmenters require extensive tuning and that the best tokeniser varies across different contexts, depending on language, downstream task, and data size. In particular, several works have highlighted the inadequacy of subword segmentation in low-resource settings and for morphologically complex languages.

- Zhu et al. [174] systematically compare different tokenisers across several tasks and languages of different morphological categories. They show that supervised morphological segmentation generally outperforms unsupervised segmenters like BPE on morphologically complex languages.

- Zhu et al. [173] demonstrate that subword segmentation is particularly beneficial for low-resource languages (because it improves performance on rare and unseen words) on tasks involving short token sequences, such as named entity recognition. They also show that BPE is outperformed by simple character n-grams across morphological complex languages (including isiZulu).

- Klein and Tsarfaty [75] show that standard tokenisation is sub-optimal for morphological analysis of Hebrew, which is a fusional morphologically rich language (morphemes undergo surface-level changes when combined into words). Performance increases when morphological information is incorporated into tokenisation and training.

- The standard approach to subword segmentation in multilingual modelling is to apply existing tokenisers (e.g. BPE and ULM) to the joint multilingual training set, resulting in a single subword vocabulary and tokeniser shared across languages. This has been shown to disadvantage low-resource languages, since it over-segment words for languages underrepresented in the multilingual training set [161, 176].

Some argue that these problems can be overcome by avoiding segmentation altogether [22] or by more sophisticated hyperparameter tuning [129]. But these limitations can be partially attributed to a more fundamental feature of current subword methods – the fact that subword segmentation operates completely separately from model training. In this setup, deep learning models for NLP are reliant on the quality of subwords produced by

segmenters. During training, model parameters are learned to optimise a training objective, but subword segmentation is fixed after preprocessing. If segmentation produces subwords that are inadequate units for modelling a particular task or language, there is no guarantee that the downstream model would be able to work around this to achieve optimal performance. No amount of training, parameterisation, or regularisation would provably overcome the barrier of fixed, and potentially flawed, subword units for modelling a language.

2.2.3 Towards Unifying Subword Segmentation and Training

Instead of applying subword segmentation during preprocessing, where it is cast as a distinct step in the modelling pipeline and completely separated from downstream training, an alternative approach is to incorporate it into model training. This strategy is a natural extension of the trend towards fully end-to-end learning which has come to dominate the field of NLP [48]. The motivation behind this would can be summarised as follows: allowing models to learn subword segmentation during training would enable them to "discover" the optimal (in terms of performance) subword segmentation scheme for a particular task and language. In this case, models would not be limited to the fixed subwords produced by BPE or ULM during preprocessing, but could instead modify the subword units on which they are trained such that it optimises their task-specific training objective.

There have been efforts towards this ideal for the task of machine translation (MT), with the goal of incorporating the modelling of subword segmentation into the architecture and training of neural MT (NMT) models.

- Following recent character-based language models [22, 147], there has been work on character-level NMT models that learn downsampled subword representations by pooling character representations [37]. However, Libovický et al. [86] find that subword NMT models still outperform their character-level counterparts.

- The Adaptive Computation Time (ACT) paradigm of Graves [51] equips sequence models with a halting unit to dynamically determine how many inputs a segment should consist of, and subsequently outputs a vector representation for the segment.

Kreutzer and Sokolov [80] extend ACT to learn source sentence segmentation during MT training, with the ACT unit learning how to group characters into subwords. They find that NMT models with source-side ACT modules prefer character-level segmentations, but perform worse than BPE and character NMT models.

- He et al. [56] introduce Dynamic Programming Encoding (DPE), which casts subword segmentation as a latent variable to be marginalised over during training. They use it to enable an NMT model to learn target sentence segmentation during training, which effectively optimises the target language subword segmentation for MT performance. Their trained NMT model is then applied as a subword segmenter, by extracting optimal segmentations using the Viterbi algorithm [157], and a standard tokenisation-based NMT model is then trained from scratch on the DPE-segmented data.

These works do incorporate aspects of subword modelling into training, but none of them fully unify neural network training and subword segmentation. The downsampling approaches [22, 147, 86, 37] are really character-level models, mainly downsampling characters to subwords for efficiency. Meanwhile, Kreutzer and Sokolov [80] limit subword segmentation learning to source sentences, while He et al. [56] limit it to target sentences. The latter also explicitly separates subword segmentation and training by using two separate NMT models – the first model, which learns subword segmentations that optimise MT performance, is used as a tokeniser for preprocessing the training data of the second MT model, which is the final NMT model. All these efforts only partially unify training and subword modelling. To move towards the development of a single model that is trained to perform some NLP task while learning to segment words into subwords, we must turn to a different paradigm of models altogether.

2.3 Segmental Sequence Modelling

The question of how raw sequential data should be segmented is important for any sequence modelling task. The technique of *segmental sequence modelling* [78, 159, 143, 70] aims to

find the segmentation strategy that will optimise sequence modelling performance, by jointly learning sequence modelling and sequence segmentation. It achieves this by incorporating the task of segmentation into the sequence model itself, such that during training the model simultaneously learns how to predict the sequence, and how to segment the sequence to improve its sequence prediction. By treating segmentation as another learnable aspect of sequence modelling, the model can adjust its segmentation scheme to optimise its training objective. The motivation behind this is that the model can then learn which segments make good units for modelling the patterns in the sequence data.

The core technical idea behind segmental sequence modelling is to cast segmentation as a latent variable and to marginalise over it during training. Given a raw data sequence $\mathbf{x} = x_1, x_2, \ldots, x_{|\mathbf{x}|}$, the probability of the sequence is modelled as

$$p(\mathbf{x}) = \sum_{\mathbf{s}:\pi(\mathbf{s})=\mathbf{x}} p(\mathbf{s}), \tag{2.3}$$

where $\mathbf{s}$ is the latent variable representing a specific segmentation of the raw sequence $\mathbf{x}$ and $\pi(\mathbf{s})$ is the unsegmented sequence corresponding to the segmentation $\mathbf{s}$ (i.e. the operation π recovers the original, unsegmented sequence). The summation is over all possible segmentations of $\mathbf{x}$, thereby marginalising over the latent segmentation variable to compute the probability of the raw sequence $\mathbf{x}$. This incorporates information about different segmentations into the probability assigned to a sequence. During training, the probabilities assigned to different segmentations will change, with the model learning to assign higher probabilities to some segmentations than others. If the training objective is to maximise $p(\mathbf{x})$, the segmentations that will end up being preferred (assigned higher probabilities) will be those that most effectively enable the accurate prediction of the sequence $\mathbf{x}$. After training, the model would still have to marginalise over all possible segmentations to compute the probability $p(\mathbf{x})$ for a raw sequence. However, the segmentation-specific probabilities $p(\mathbf{s})$ computed for the marginalisation would reveal how different segmentations are ranked in

terms of assigned probabilities, with the optimal segmentation being

$$\mathbf{s}^* = \underset{\mathbf{s}:\pi(\mathbf{s})=\mathbf{x}}{\operatorname{argmax}}\, p(\mathbf{s}), \tag{2.4}$$

where $\mathbf{s}^*$ denotes the segmentation that maximizes the probability among all possible segmentations of the sequence $\mathbf{x}$.

In this way, training a sequence model to optimise Equation 2.3 allows the model to "discover" the optimal segments for sequence modelling. This leaves decisions around sequence segmentation to the model itself, thereby fully unifying sequence modelling and sequence segmentation in a single end-to-end trainable model. Some of these segments may align with natural sequence units, like words in text or phonemes in speech, as these segments represent meaningful units that form the basis of predictable patterns in the sequences i.e. preferring these units might be beneficial for sequence modelling.

Segmental sequence modelling, as outlined above, is a very general class of generative models characterised by modelling segmentation as a latent variable and computing the probability of a sequence by marginalising over all its valid segmentations. The distribution $p(\mathbf{s})$ can be instantiated by any probabilistic model and the marginalisation can be computed, or approximated, through several possible algorithms.

A few variants of segmental sequence modelling have been instantiated as neural sequence models. Kong et al. [78] propose a bidirectional RNN that learns segment embeddings for handwriting recognition and part-of-speech tagging. Wang et al. [159] propose SWAN (Sleep-WAke Networks), a segmental RNN for text segmentation and speech recognition. Both of these models use dynamic programming to efficiently compute the marginal likelihood during training and to find the optimal segmentation of a sequence.

Sun and Deng [143] were the first to adapt segmental sequence modelling for neural language modelling, coining the term "segmental language model" (SLM). Their SLM is a generative model that generates text as a sequence of multi-character segments. At each position in a text sequence, they use one LSTM to encode the text up to that point, and another to generate the next segment. Each segment is generated one character at a time, followed by

a special end-of-segment character. They also use a dynamic programming algorithm during training to compute the likelihood of a text sequence in linear time complexity. They train their SLM on Chinese language modelling and evaluate their learned SLM segmentations on the task of unsupervised Chinese word segmentation. This is an important task in Chinese NLP, since Chinese sentences have no explicit word boundaries. The motivation behind applying a segmental sequence model to this problem is that it might learn segment boundaries that resemble word boundaries, since words are meaningful units. This is supported by their results, where SLM performs competitively as an unsupervised Chinese word segmenter.

The SLM was extended by Kawakami et al. [70], who introduced two additional features. Firstly, they equip the model with a lexical memory – a vocabulary of segments that it can use to generate a segment as a single event (instead of one character at a time). The idea behind this is to store high-frequency segments in a lexicon and to model their probabilities directly. Secondly, they introduce an expected length penalty, which acts as a regulariser during training to prevent bias towards generating long segments. They also apply their SLM as an unsupervised Chinese word segmenter, outperforming other unsupervised approaches. Furthermore, they show that their SLM is able to partially recover word boundaries in English text in which whitespaces between words have been removed.

Segmental sequence models for unsupervised word segmentation have also been proposed as Transformer-based masked language models [32] and LSTM-based bi-directional language models [160]. These works are the most recent contributions to a growing body of work on segmental sequence modelling in NLP. The focus of this research has been on analysing whether segmental sequence modelling can learn to segment raw text into words, without having access to word boundaries. This line of research has confirmed two findings. Firstly, that segmental sequence modelling can be successfully instantiated as neural architectures (LSTMs, Transformers) and used for several sequence modelling tasks (autoregressive language modelling, masked language modelling, bi-directional modelling). Secondly, without explicit supervision, segmental sequence models learn to segment text into linguistically plausible units of text – to some extent they "discover" words as the primary unit of meaning.

While word segmentation is an important problem for Chinese NLP, it is not critical for languages with explicit word boundaries. However, the success of SLMs for word discovery does raise a practically relevant question: can segmental sequence modelling be used to model the segmentation of words into subwords? So far segmental sequence models have been used to optimally segment handwritten words, speech data, and Chinese words, but they have not been used to discover subword units. That is the method explored in this thesis – to adapt segmental sequence modelling for subword segmentation.

2.4 Text Generation for Low-Resource Languages

There is a growing body of research on improving neural text generation models for low-resource languages [57]. Most of the existing work has focused on data augmentation techniques or cross-lingual transfer learning [54], as opposed to modelling innovations.

2.4.1 Task-Specific Inductive Biases

From a modelling perspective, a common approach is to modify vanilla neural architectures or standard training objectives to incorporate task-specific inductive biases. These modifications aim to encode knowledge about the task at hand, which can be leveraged for more sample-efficient training. This is usually achieved by imposing additional structure onto models. As a result, structured text generation tasks, like sequence-to-sequence generation, are particularly exploitable in this way.

NMT for low-resource languages has been improved by equipping models with the ability to access resources that aid translation. Memory-augmented NMT [41] retrieves information from the training corpus during testing, helping with rare word translation and improving low-resource performance [14]. Similarly, the use of bilingual lexicons can aid word-level translation [162]. Akyurek and Andreas [5] equip their NMT model with a lexical translation mechanism, allowing it to leverage word-level translation mappings and boosting low-resource performance. Other structured generation tasks have benefited from explicitly separating the generation problem into task-relevant subproblems. In data-to-text generation,

Ma et al. [88] show that modelling the content of generations, through intermediary generation, substantially improves low-resource performance. Another effective approach is to design more expressive training objectives. In low-resource dialogue generation, introducing latent variables to separately model the underlying structure and surface text has shown to be effective [151, 170].

2.4.2 Language-Specific Inductive Biases

A related approach is to encode language-specific inductive biases, which leverage the linguistic properties of specific languages for more sample-efficient training. This is usually done by incorporating linguistic annotations into modelling. Earlier in this Chapter (in Section 2.2.1) we mentioned one way of doing this, by incorporating morphological information into subword tokenisation. Alternatively, linguistic information can be injected into the model architecture to improve performance for low-resource languages.

A recent example of this is KinyaBERT [103], a masked language model for the Rwandan language Kinyarwanda with a two-tier neural architecture that incorporates a morphological analyser. Kinyarwanda is a morphologically rich language, so subword information is especially critical and existing subword segmenters are sub-optimal. Their modified architecture leads to performance gains over vanilla masked language model arcitectures. They inject linguistic information via a hand-crafted morphological analyser, which would have to be developed to adapt their architecture to new languages.

The practical application of this thesis is to build better text generation models for the Nguni languages. They are a group of related South African languages with highly agglutinative morphological systems. In adapting segmental sequence modelling for subword segmentation, we aim to leverage this linguistic knowledge by explicitly modelling the complex subword structure of Nguni languages. Our approach is motivated by the same factors as KinyaBERT. Given the morphological complexity of the Nguni languages, they are ideal candidates for benefiting from optimised subword segmentation. In contrast to approaches like KinyaBERT, which require linguistic tools or annotations, our technique is an end-to-end model that requires no external resources. In this sense, we position our approach

as linguistically informed data-driven modelling – developing neural network architectures that, without the injection of linguistic knowledge, target the linguistic properties of particular languages. We encode language-specific inductive biases in the design of our subword segmental architecture and training objective.

2.4.3 Multilingual Modelling

Such linguistically informed modelling is quite rare in modern NLP research. Most efforts in low-resource NLP are centered around transfer learning strategies and data augmentation techniques [57]. Currently the most successful approach to low-resource NLP is *multilingual modelling* – training a single model on multiple languages simultaneously. This does not require architectural innovations – standard neural network architectures (e.g. Transformers, encoder-decoder models) are simply trained on a collated dataset that includes multiple languages. This leads to improved performance on individual languages through *cross-lingual transfer*. Some of the knowledge gained about one language (e.g. syntactic rules, semantic relationships between words) can transfer to knowledge about other languages, given underlying similarities between languages. This is especially beneficial for low-resource languages, since they can experience positive transfer from high-resource languages with much larger datasets [53, 4].

Multilingual modelling is in some ways orthogonal to linguistically informed modelling, since they can be applied independently or combined in a single model. However, they do exemplify the most popular current approach to the problem we are trying to solve, which is to improve text generation models for the low-resource Nguni languages. As such, we now survey some of the trends in multilingual modelling, highlighting existing works which have included Nguni languages.

Massively Multilingual Modelling

Multilingual pretrained language models (PLMs) have revolutionised NLP for low-resource languages. It enables cross-lingual transfer from high-resource languages, as demonstrated by encoder-only models like mBERT [29] and XLM-R [23], and encoder-decoder models

like mT5 [169] and ByT5 [168]. Models are now pretrained on over 100 languages, reflecting a trend towards increasingly multilingual PLMs. A few massively multilingual PLMs (more than 100 languages) include isiXhosa or isiZulu in their pretraining, but not isiNdebele nor Siswati. Among encoder-only PLMs, XLM-R [23] includes isiXhosa. Among encoder-decoder PLMs, mT5 [169] and ByT5 [168] include isiXhosa and isiZulu. None of these works evaluate their model on any of the Nguni languages.

While increasing multilinguality has undoubtedly benefited low-resource languages, it also has certain shortcomings. A major limitation is the lack of proper multilingual evaluation. Massively multilingual PLMs might be trained on over 100 languages, but it is not possible to perform downstream evaluation for all these languages since many lack high-quality evaluation datasets. As a result, the full extent of the low-resource language capabilities of multilingual PLMs are unknown.

Multilingual African Language Modelling

Another limitation of scaling PLMs to more languages is that it sacrifices greater gains on individual languages for gains averaged across several languages. While models like XLM-R and ByT5 do achieve impressive results on a broad spectrum of languages, PLMs focused on smaller groups of languages can outperform more multilingual models on the targeted languages [6, 28, 34]. The strategy of increasing multilinguality is not optimal if we are focused on improving performance for specific low-resource languages.

In response to this, a number of works have trained multilingual PLMs for a more limited number of African languages. AfroLM [31] is trained from scratch with active learning on 23 African languages, including isiXhosa and isiZulu. Afro-XLMR [6] adapt XLM-R for 17 African languages, including isiXhosa and isiZulu. They achieve this through multilingual adaptive finetuning, which entails continuing the pretraining of an existing PLM on a corpus of selected languages. Adelani et al. [1] use the same approach for encoder-decoder PLMs, where adaptation amounts to continued span denoising training. They adapt mT5 and ByT5 for 17 African languages, including isiXhosa and isiZulu, to produce respectively AfriMT5 and AfriByT5. Their work is focused on MT (including isiZulu $\leftrightarrow$ English), so they do not

evaluate these models on other NLG tasks. In the MT experiments, they find that multilingual adaptation improves performance and that AfriByT5 outperforms AfriMT5. Afro-XLMR and AfriByT5 represent the most successful efforts to date for developing pretrained language models for the Nguni languages, albeit only isiXhosa and isiZulu.

3

Subword Segmental Language Modelling

In this chapter we propose our first subword segmental model, for the task of language modelling. The subword segmental language model (SSLM) is an LSTM-based decoder-only model that jointly learns subword segmentation and autoregressive (left-to-right) language modelling. During training, the SSLM adapts its subword segmentation scheme to optimise its language modelling training objective. We compile relatively small but high-quality language modelling datasets for all four Nguni languages. We train an SSLM for each Nguni language and evaluate our models on held-out test corpora using a perplexity-based metric. SSLM outperforms tokenisation-based language models on average across the Nguni languages. SSLM also outperforms standard subword tokenisers on the task of unsupervised morphological segmentation, demonstrating that its learned subwords are closer to morphemes, To explore the full potential of subword segmental modelling for unsupervised morphological segmentation, we also train word-level SSLMs. These models comfortably outperform popular unsupervised segmenters, establishing SSLM as a strong alternative. SSLM is the first practical implementation of subword segmental modelling, so it required extensive experimentation and hyperparameter tuning. We report our findings in this regard and present an analysis of the role of different subword segmental components on model performance.

3.1 Motivation

In some sense, language modelling represents the most general way of building computational models of language. Language models (LMs) are trained to predict the identity of concealed words, based on either the surrounding context (*masked* language modelling) or the preceding context (*left-to-right* or *autoregressive* language modelling). Succeeding at this task depends on modelling the statistical patterns of word co-occurrence across a corpus, which are in turn determined by the syntax and semantics of natural language. Language modelling therefore serves as the ideal testbed for assessing how effectively a statistical model can learn a language. The aim of this thesis is to develop text generation models, so we focus on autoregressive language modelling. This can be framed as a sequence modelling task – the goal is to continually predict the next word in a sentence or document, given the preceding sequence of words. Autoregressive pretrained language models (PLMs) trained on sufficiently large text corpora can generate plausible continuations of incomplete input text or can be finetuned for specific text generation tasks.

Early PLMs like BERT [29] and GPT-2 [122] were trained on text corpora segmented into subwords during preprocessing, by tokenisers like BPE [135] and WordPiece [133]. This approach has proved so successful that remains standard practice in more recent LLMs like LLaMa [150]. As a result, these LMs are really subword-level LMs – they are trained to predict the next subword token given the preceding sequence of subword tokens. When these LMs are used for generation, they generate text one subword token at a time.

Finalising subword segmentation during preprocessing might be adequate given a sufficiently large training corpus. A model exposed to enough variation in subword usage could learn robust representations, irrespective of the choice of subword units. However, this is not guaranteed for low-resource languages, where training might be significantly compromised by fixed subword segmentation. Given limited training data, the model might not be able to overcome suboptimal subword units to learn generalisable features. This is related to the out-of-vocabulary problem, which affects low-resource languages disproportionately. Available datasets are small, so any held-out dataset will contain rare or new words. There-

sesihambe	
Morphemes	se–si–hamb–e
BPE	sesi–ha–mbe
Unigram LM	se–si–hambe
Morfessor	se–s–ihambe
SSLM	se–si–hamb–e

Table 3.1: Segmentations of the isiXhosa word *sesihambe* produced by subword tokenisers, compared to the actual morphemes and the maximising segmentation of our model (SSLM).

fore it is critical for models to learn subword features that enable them to effectively model morphological composition.

The Nguni languages are both under-resourced and morphological complex, which further exacerbates the problem of suboptimal subword segmentations. Because of their agglutinative morphology and conjunctive orthography, words are far too coarse grained as units for training LMs. Morphemes are the primary linguistic units and words are formed by concatenating morphemes. For example, the isiXhosa word "sesihambe", morphologically segmented as "se–si–hamb–e", means "we are gone", where "se" means "we", "si" means "are", and "hamb–e" means "gone", with the "–e" suffix indicating past tense. Since these morphemes are written conjunctively (without orthographic boundaries, whitespaces, between them), subword segmentation is required to discover the underlying linguistic units that would be required for language modelling. As shown in Table 3.1, existing segmenters do not reliably capture morphological boundaries, which prevents morphological composition (composing the meaning of a word from its constituent morphemes) and could in turn impede effective language modelling.

Subword tokenisers like BPE [135] and WordPiece [133] build subword vocabularies based on subword frequencies. This produces efficiently segmented text corpora, but does not guarantee good subword units for language modelling. Ideally, a subword segmentation scheme should be tailored to improve LM performance. While ULM [81] and Morfessor [25] are based on probabilistic models, they do not incorporate the context surrounding a word into their tokenisation algorithms. Their models operate under the assumption that a word can be segmented independently of its context. This simplifying assumption leads to computationally

efficient algorithms, but conflicts with the inherent context dependence assumed by LMs, which compute the probability of the next word conditioned on the preceding context. Furthermore, these tokenisers and their probabilistic models function entirely independently of the eventual LM. Subword segmentation is finalized during preprocessing, and the LM is then trained from scratch on fixed subword tokens. As a result, there remains a disconnect between tokenisation algorithms and LM training, which introduces the mismatch between the objectives of subword segmentation and language modelling in the first place.

In this chapter we propose the subword segmental language model (SSLM), which learns how to segment words while training as an autoregressive LM. The SSLM adapts subword segmentation to optimise a left-to-right language modelling objective. In line with language modelling assumptions, the resulting subword segmentation scheme is context-dependent (the preferred subword segmentation of a word will depend on the preceding context). This allows our model to learn the subword units that it can most effectively leverage for language modelling. SSLM represents a radical departure from tokenisers like BPE and ULM, which cast subword segmentation as a context-independent preprocessing step. SSLM trains subword segmentation and language modelling jointly, so it can learn subwords that optimise conditional LM generation. In this chapter we show that this proves beneficial for handling the data scarcity and morphological complexity of the Nguni languages.

3.2 Generative Model

SSLM generates a sequence of space-separated words $\mathbf{w} = \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n$, corresponding to an underlying sequence of characters $\mathbf{x}$, and generates each word $\mathbf{w}_i$ as a sequence of subwords $\mathbf{s}_i = s_{i1}, s_{i2}, \ldots, s_{i|\mathbf{s}_i|}$, where s_{ij} is the j^{th} subword of the i^{th} word. The probability of a text sequence $\mathbf{w}$ is computed by marginalising over all possible word segmentations as

$$p(\mathbf{w}) = \sum_{\mathbf{s}:\pi(\mathbf{s})=\mathbf{w}} p(\mathbf{s}), \tag{3.1}$$

where $\pi(\mathbf{s})$ is the original, unsegmented sequence of words underlying the segmented sequence of subwords $\mathbf{s} = \mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n$. Using the chain rule, we define the probability of a particular subword segmentation $\mathbf{s}$ as

$$p(\mathbf{s}) = \prod_{i=1}^{|\mathbf{w}|} \prod_{j=1}^{|\mathbf{s}_i|} p(s_{ij} | \mathbf{s}_{\leq i, < j}), \tag{3.2}$$

where $\mathbf{s}_{\leq i, < j}$ is the subword sequence preceding the j^{th} subword of the i^{th} word (this includes all subwords in the preceding words and the subwords preceding s_{ij} in the current word).

We treat white spaces and punctuation as assumed segments, equivalent to 1-character words. In this way we implicitly model word boundaries. Predicting a non-alphabetical character (e.g. space) signals a word ending. Segments cannot cross word boundaries, so the only segmentation learned by the model is how to segment words into subwords.

3.3 Dynamic Programming Algorithm for Training

Conditioning the probabilities of a segment $p(s_{ij} | \mathbf{s}_{\leq i, < j})$ on all possible segmentation histories is computationally intractable, so we follow previous segmental sequence models [78, 70] by introducing a conditional semi-Markov assumption,

$$p(s_{ij} | \mathbf{s}_{\leq i, < j}) \approx p(s_{ij} | \pi(\mathbf{s}_{\leq i, < j})) = p(s_{ij} | \mathbf{x}_{< k}), \tag{3.3}$$

where $\mathbf{x}_{< k}$ is the raw, unsegmented character sequence preceding the current segment (assuming the current segment starts at the k^{th} character). The segment generation probability does not depend on segmentation of the preceding sequence of words, or within the current word. Instead, the probability is conditioned on the unsegmented character history. This enables us to compute the marginal likelihood of Equation 3.1 incrementally using a dynamic programming algorithm. Given $\alpha_0 = 1$, at each step the algorithm computes a forward score,

$$\alpha_t = \sum_{k=f(\mathbf{x},t)}^{t} \alpha_k p(s = \mathbf{x}_{k:t} | \mathbf{x}_{< k}), \tag{3.4}$$

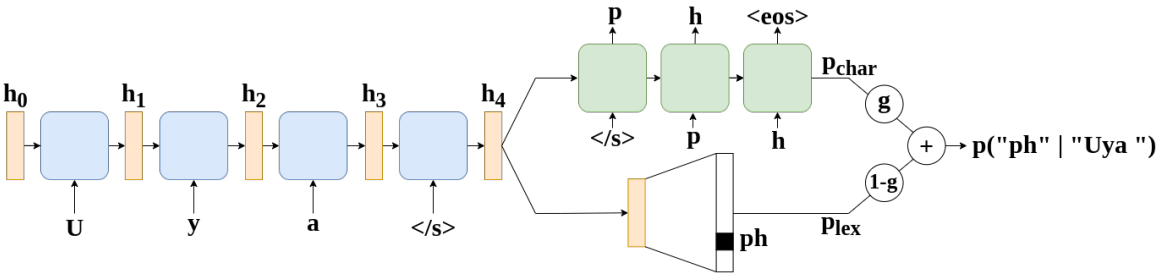

Figure 3.1: The SSLM computing the probability for the subword segment "ph" in the isiXhosa sentence "Uya phi?" A character-level LSTM encodes the unsegmented text history "Uya ", while a mixture model (Equation 3.5) that interpolates between a character-level LSTM decoder and a lexicon model generates the segment "ph". This is repeated for all possible subwords in a sequence to compute the forward scores of Equation 3.3.

where $f(\mathbf{x}, t)$ returns the starting index of the current word, since the longest possible subword segment ending at index t extends to the start of the current word.[1] Each of the expressions in the summation computes the probability of concluding the sequence at character index t by generating a segment starting at index k. The forward score α_t represents the probability of generating the sequence up to the character x_t, marginalised over all possible subword segmentations up to that point. We can efficiently compute the marginal of the full character sequence, Equation 3.1, as $p(\mathbf{w}) \approx p(\mathbf{x}) = \alpha_{|\mathbf{x}|}$.

3.4 Neural Architecture

The dynamic programming algorithm outlined above is dependent on modelling the distribution $p(s = \mathbf{x}_{k:t} | \mathbf{x}_{<k})$ in Equation 3.4, which is the probability assigned to the generation of an arbitrary subword segment $\mathbf{x}_{k:t}$ given the preceding unsegmented character sequence $\mathbf{x}_{<k}$. We parameterise this distribution with a neural sequence model augmented with subnetworks for subword generation, which forms the architecture of our SSLM (visualised in Figure 3.1).

[1] In practice, we limit the longest possible subword to a fixed character length and ignore segments exceeding this. This discards a subset of possible subword segmentations, but greatly improves computational efficiency. The maximum subword segment length is a prespecified hyperparameter, discussed in the next subsection.

As shown in Figure 3.1, each segment probability is computed as a mixture of two probability distributions as

$$p(s_{ij}|\mathbf{x}_{<k}) = g_k p_{\text{char}}(s_{ij}|\mathbf{x}_{<k}) + (1-g_k)p_{\text{lex}}(s_{ij}|\mathbf{x}_{<k}), \tag{3.5}$$

$$= g_k p_{\text{char}}(s_{ij}|\mathbf{h}_k) + (1-g_k)p_{\text{lex}}(s_{ij}|\mathbf{h}_k) \tag{3.6}$$

where $\mathbf{h}_k$ is the hidden state of a character-level LSTM that encodes the sequence history $\mathbf{x}_{<k}$, g_k is a mixture coefficient, p_{lex} is a fully connected neural layer that generates the entire segment from a lexicon as a single event, while p_{char} is an LSTM decoder that generates the segment character by character. The probability assigned by p_{lex} is produced by a softmax function over the subword lexicon, while p_{char} is computed using the chain rule over the subword character sequence as

$$p_{\text{char}}(s|\mathbf{x}_{<k}) = \prod_{n=k}^{k+|s|} p(x_n|\mathbf{x}_{<n}) \times p(\texttt{<eos>}|\mathbf{x}_{<k+|s|}), \tag{3.7}$$

where subword s consists of the characters $x_k, x_{k+1}, ..., x_{k+|s|}$ and <eos> is a special end-of-segment character, all of which are assigned probabilities by a character-level LSTM decoder (the subnetwork visualised by green cells in Figure 3.1). The hidden state of this LSTM decoder is initialised with $\mathbf{h}_k$, the sequence history encoding, and the subsequent character generations in the subword are conditioned on preceding characters via the character decoder's updated hidden state (the hidden state of the green subnetwork in Figure 3.1).

The character and lexicon models are conditioned on the unsegmented text history through $\mathbf{h}_k$, a vector representation computed by a character-level LSTM encoder. The LSTM hidden state $\mathbf{h}_k$ represents the entire sequence history. This is passed to segment prediction subnetworks, namely our lexicon and character-level models, to achieve the segmentation-invariant conditioning required by Equation 3.4. The conditioning mechanism is visualised for $\mathbf{h}_4$ in Figure 3.1. For the lexicon model, the conditioning is achieved by passing $\mathbf{h}_k$ as input to its fully connected layer. For the character-level model, the conditioning is achieved by initialising the LSTM subnetwork hidden state with $\mathbf{h}_k$. The mixture coefficient

g_k is also computed from $\mathbf{h}_k$ with a fully connected and trainable neural layer, so the model can dynamically weigh the influence of the lexicon and character-level subnetworks via the mixture weights. This allows the SSLM to learn when to rely on the lexicon and when to revert to character-by-character generation.

The lexicon is a subword vocabulary constructed during preprocessing and fixed during training. It consists of the V most frequent subwords up to L characters long (excluding segments that contain any non-alphabetical characters).[2] Both V (lexicon size) and L (maximum subword segment length) are pre-specified hyperparameters that determine the makeup of the resulting lexicon. The lexicon model p_{lex} produces a conditional distribution over frequent subwords, allowing SSLM to directly model commonly occurring subwords. The dynamically weighted mixture model of Equation 3.5 allows SSLM to assign non-zero probabilities to any subword unit (so the model is not limited to its lexicon) since the character-level model p_{char} can generate new subwords as character sequences. During training, SSLM can learn to switch between the lexicon for common subwords (e.g. morphemes) and character-level generation for rare segments (e.g. named entities).

The resulting architecture is a fully parameterised and end-to-end trainable neural sequence model. It can be trained, using any neural optimisation technique, to maximise the log likelihood over a training corpus. The likelihood of a batch of sequences can be computed iteratively and efficiently with the aid of Equation 3.4.

3.5 Experimental Setup

We implement SSLM as a model in the open-source fairseq library [107] and publicly release all our data, code, and trained models.[3] We now detail our procedure for training SSLMs, including the data collection process, hyperparameter tuning, and our training setup.

[2] During development, we experimented with populating the subword lexicon with morphemes extracted from a morphologically annotated corpus, to instill prior knowledge of morphological units. This degraded LM performance, so we reverted to a frequency-based lexicon. Given that most affixes (non-stem morphemes) occur frequently, most affixes will be included in a lexicon based on subword frequency, given a sufficiently large and representative corpus.

[3] https://github.com/francois-meyer/subword-segmental-lm

Language	Size (# words) Train	Valid/Test	Type	Source
isiXhosa (xh)	**3.4mil**	**190k**		
– NCHLT Text	1.1m		monolingual	government websites
– SADiLaR Monolingual	2.4m		monolingual	government websites
– Navy Corpus	500k		parallel	government websites
isiZulu (zu)	**3.1mil**	**200k**		
– NCHLT Text	1.5m		monolingual	government websites
– Autshumato	400k		parallel	government websites
– Isolezwe News Corpus	1.8m		monolingual	news articles
isiNdebele (nr)	**450k**	**25k**		
– NCHLT Text	750k		monolingual	government websites
Siswati (ss)	**500k**	**28k**		
– NCHLT Text	800k		monolingual	government websites

Table 3.2: Our language modelling datasets are compiled from these publicly available corpora. For the individual data sources, we list the raw data sizes. As a result of data cleaning, the compiled train/valid/test sets are smaller than the combined source corpora.

3.5.1 Data

We train our models on language modelling datasets that we compile ourselves for isiXhosa (**xh**), isiZulu (**zu**), isiNdebele (**nr**), and Siswati (**ss**). For each language we collect free text datasets publicly available at the time of research, and combine them into a single corpus. The composition of these corpora are summarised in Table 3.2.

To avoid the pitfalls of low-resource data collection [79], we set specific criteria for including datasets. We collect datasets from reputable sources such as the South African Centre for Digital Language Resources (SADiLaR).[4] We also inspect individual datasets (manually and automatically with text processing scripts) and discard datasets of questionable quality (e.g. containing a significant amount of English text). Following this process, we were left with the following data sources:

[4]https://repo.sadilar.org/discover

- The NCHLT Text Corpora [38] contains monolingual unannotated corpora for ten South African languages. The data is scraped from South African government websites.
- The Autshumato dataset[5] is a collection of parallel corpora for English and several South African languages. We use the isiZulu side of the English–isiZulu parallel corpus, which was scraped from South African government websites.
- The SADiLaR Monolingual isiXhosa corpus[6] is an expanded version of the isiXhosa side of the English–isiXhosa Autshumato parallel corpus, scraped from South African government websites.
- The OPUS isiXhosa Navy Corpus [149] is an English–isiXhosa parallel corpus consisting of documents produced by the South African navy. We use the isiXhosa sentences of the dataset.
- The Isolezwe News Corpus[7] contains news articles in isiZulu, scraped from the online isiZulu newspaper, Isolezwe.[8]

Where there is overlap between data sources, we remove duplicated text to prevent repetitive text and leakage between training and held-out data. Following preprocessing and cleaning, we split the individual corpora 80%/10%/10% into train/validation/test sets before combining them respectively into one train/validation/test data set. This ensures that the individual corpora are equally distributed in the training and evaluation sets. We end up with a collection of datasets that include publicly available monolingual corpora and monolingual data extracted from machine translation datasets. The sizes of individual source corpora, as well as our final train/validation/test sets, are listed in Table 3.2. The isiXhosa and isiZulu datasets are much larger than the isiNdebele and Siswati datasets, owing to a difference in the public availability of data for these languages. Consequently, we view this work as investigating two levels of *low-resourcedness*. We assess the value of SSLM for relatively

[5]https://repo.sadilar.org/handle/20.500.12185/399
[6]https://repo.sadilar.org/handle/20.500.12185/524
[7]https://github.com/newstools
[8]https://www.isolezwe.co.za/

Hyperparameter	Grid search range	SSLMs	LSTM baselines	Transformer baselines
Attention heads	4, 8		4/8*	
Layers	1, 3, 5	3	3	3
Embedding size	128, 512	512	128/512*	512
Hidden size	128, 512, 1024	1024	1024	1024
Learning rate	1e-2, 1e-3, 1e-4	1e-3	1e-3	1e-3
Dropout	0.1, 0.2, 0.5	0.5	0.2	0.1
Batch size	64, 128	64	64	64
Sequence length		120 chars	120 chars	120 chars
Weight decay	1e-5	1e-5	1e-5	1e-5
Gradient clip	1.0	1.0	1.0	1.0

Table 3.3: Hyperparameter settings for all our models, with * indicating where optimal values (based on validation BPC) depended on the language. For embedding size, 128 was optimal for isiXhosa and Siswati, while 512 was optimal for isiZulu and isiNdebele. The Transformer models had 8 attention heads for isiXhosa and isiZulu, and 4 for isiNdebele and Siswati.

low-resource languages like isiXhosa and isiZulu, compared to extremely low-resource languages like isiNdebele and Siswati.

3.5.2 Hyperparameters

We tune the training hyperparameters for SSLM on isiXhosa language modelling and use these settings for the other three Nguni languages. We run a grid search over the hyperparameter ranges shown in Table 3.3, selecting the values that optimise perplexity on our isiXhosa validation set. We use the Adam optimiser during training [74], halving the learning rate if validation loss failed to improve for 3 epochs and stopping when no validation loss improvement occurred for 6 epochs. We employ several standard regularisation techniques, including dropout (on all except recurrent layers), weight decay, and gradient clipping. Table 3.3 shows the hyperparameter settings we settled on for our SSLMs after tuning.

Subword segmental modelling also introduces the need to specify a maximum subword segment length (to make Equation 3.4 computationally feasible) and the size of the subword lexicon. During development, we found that changing these hyperparameters markedly affected validation performance. We tune these values separately for each lan-

Model	Lexicon size	Maximum subword segment length
isiXhosa	10k	5
isiZulu	10k	5
isiNdebele	5k	10
Siswati	10k	20

Table 3.4: Lexicon hyperparameter values for our SSLMs.

guage by optimising validation set perplexity, varying the lexicon size through the range $\{1\text{k}, 10\text{k}, 20\text{k}, 30\text{k}\}$ and the maximum segment length across $\{5, 10, 15, 20\}$. The optimal lexicon size and maximum segment length are listed in Table 3.4, which shows that they varied across the languages and resource levels. We discuss these effects and the role of these hyperparameters in more detail in Section 3.8.

3.5.3 Training

We train our models on virtually partitioned instances of NVIDIA A100 GPUs with 3 compute units[9] and 20GB memory. The isiXhosa and isiZulu long-range SSLMs converged after about 40 epochs of the training corpus, taking 3 days to train. The isiNdebele and Siswati long-range SSLMs converged after 30 epochs, taking 10 hours to train.

3.6 Experiment 1: Intrinsic Language Modelling

We evaluate our trained SSLMs by computing an intrinsic measure of language modelling performance on a held-out test set. We compare the intrinsic performance of SSLM to tokenisation-based baselines, allowing us to assess whether learning subword segmentation during training improves the predictive capabilities of an LM.

[9]The full GPU consists of 7 compute units, so we trained on $\frac{3}{7}$ of the GPU.

3.6.1 Baselines

For each language we train an SSLM and six tokenisation-based LMs as baselines. Our baselines include three standard subword methods: character tokens, BPE, and ULM. For each tokeniser we train LSTM and Transformer LMs. As in the case of our SSLMs, we tune the hyperparameters of our baselines by optimising for validation performance (BPC). The final hyperparameter settings for our baselines are provided in Table 3.3.

3.6.2 Evaluation Metric

The intrinsic evaluation metric we use to compare LMs is bits-per-character (BPC). Like perplexity, it measures the quality of an LM by quantifying how well it predicts a sample, with lower BPC scores indicating a better fit. We use BPC as our evaluation metric, instead of perplexity, because it can be compared across models with different underlying subword segmentations [50]. It is cross-entropy-based and normalised by character length, which enables a fair comparison between character-based models, tokenisation-based models, and SSLM. BPC is computed as

$$\text{BPC}(X) = -\frac{1}{N}\sum_{\mathbf{x}\in X}\log_2 p(\mathbf{x}), \tag{3.8}$$

where X is a corpus of text sequences $\mathbf{x}$, which are represented by tokens of any type (e.g. characters, subwords, or words), and N is the length of the corpus in characters. We compute BPC on our LM test sets to evaluate generalisation to a held-out corpus.

3.6.3 Results

BPC scores on the validation and test sets are shown in Table 3.5. SSLM emerges as the best LM across the Nguni languages, outperforming all tokenisation-based LMs on average. It also achieves the best test set BPC for each language individually, except isiXhosa, where is second only to the ULM-based LSTM. For the Nguni languages, unifying language modelling

	Validation set BPC					Test set BPC				
Model	**xh**	**zu**	**nr**	**ss**	*avg*	**xh**	**zu**	**nr**	**ss**	*avg*
Char-LSTM	1.24	1.22	1.41	**1.38**	1.31	1.32	1.26	1.39	1.30	1.32
BPE-LSTM	1.23	**1.19**	1.39	**1.38**	1.30	1.30	1.22	1.39	**1.28**	1.30
ULM-LSTM	**1.22**	1.23	1.39	1.40	1.31	**1.25**	1.27	1.39	1.31	1.31
Char-Transformer	1.51	1.43	1.49	1.49	1.48	1.53	1.48	1.47	1.43	1.48
BPE-Transformer	1.30	1.22	1.38	**1.38**	1.32	1.33	1.27	1.36	1.30	1.31
ULM-Transformer	1.32	1.22	1.38	**1.38**	1.35	1.34	1.27	1.36	1.29	1.31
SSLM	1.24	**1.19**	**1.35**	**1.38**	**1.29**	1.27	**1.21**	**1.35**	**1.28**	**1.28**

Table 3.5: Intrinsic LM performance, as measured by validation and test set BPC scores.

and subword segmentation leads to a model that learns how to segment words such that its LM predictions are more accurate than tokenisation-based LMs.

Among our baselines, the best neural architecture and subword tokeniser depends on the language. For example, Transformer models outperform LSTMs for isiNdebele, but perform poorly for isiXhosa. There is also substantial variance in the relative performance of characters, BPE, and ULM on different languages. This inconsistency is one of the main limitations of current subword tokenisers that we are trying to address with subword segmental modelling. The results show that the SSLM succeeds in this regard – it not only outperforms baselines on average across languages, but is also more consistent when considering individual language performance. We attribute this to SSLM's ability to optimise subword segmentation individually for each language, making it more robust to the particularities of different languages and, therefore, more broadly applicable.

3.7 Experiment 2: Morphological Segmentation

To assess the morphological knowledge acquired by SSLM during training, we evaluate its learned subword segmentation scheme on the task of unsupervised morphological segmentation (UMS). The goal in this task is to develop systems that accurately predict morpheme boundaries, without supervised training on morphologically annotated data. This is a challenging NLP task for morphologically rich languages [114, 40, 153]. Our aim is to test

whether SSLM is learning linguistically plausble subword segmentations, by measuring to what extent it discovers morphemes as linguistic units.

3.7.1 Data

We evaluate our models on morphologically annotated data from the Annotated Text Corpora[10] released by the National Center for Human Technology (NCHLT) in South Africa [38]. The data consists of train and test sets of morphologically analysed text for South African languages. In the dataset, words are segmented into their *canonical segmentations* i.e. standardised morphemes that do not necessarily correspond to word substrings [24]. For example, the canonical segmentation of the isiXhosa word "yedwa" is "ya-i-dwa" (the morphemes are fused in the word form). SSLM segments words into substrings, so we require *surface segmentations* (segments correspond to substrings) for evaluation. Most of the segmentations can be used as is, because the canonical and surface segmentations are identical. Where segmentations diverge, we use the scripts made available by Moeng et al. [98] to map canonical segmentations to surface segmentations. They employ a heuristic approach based on the Levenshtein distance minimal edit operations to map from the canonical form to the surface form. They also filter out tokens that are unsuitable for morphological segmentation (e.g. punctuation, named entities). We evaluate our models on the test sets for isiXhosa, isiZulu, isiNdebele, and Siswati. Each test set consists of 3,500 morphologically analyzed words occurring in the context of complete sentences, so the surrounding context can be taken into account by segmenters.

3.7.2 Models

Subword segmental models To apply SSLM as a segmenter, we use the Viterbi algorithm to compute the segmentation that maximises the likelihood of a sentence according to a trained SSLM. This extracts the subword segmentation that the model has learned to assign the highest probability to. Segmenting the morphological evaluation data in this way takes

[10]Datasets are available at https://repo.sadilar.org/handle/20.500.12185/7

Model	Lexicon size	Maximum subword segment length
isiXhosa	10k	10
isiZulu	5k	20
isiNdebele	10k	10
Siswati	5k	20

Table 3.6: Lexicon hyperparameter values for our subword segmental word models (SSWMs).

less than a few minutes on a personal computer, since the Viterbi algorithm is computationally efficient. For each language, we evaluate two subword segmental models. The first group of models are the SSLMs trained in Section 3.6 for long-range language modelling.

Secondly, we introduce a new set of models trained on single words in isolation, which we call subword segmental word models (**SSWMs**). These subword segmental models are not LMs. They are trained with the same architecture, training algorithm, and on the same training data as our long-range SSLMs, but they process one word at a time without any surrounding context. This removes the need to model long-range linguistic dependencies, allowing the subword segmental model to focus on the short-range task of word prediction and segmentation. We developed these models because it is common for UMS algorithms (e.g. Morfessor) to operate on the word-level. Our aim is to investigate the full potential of subword segmental modelling for UMS, uncoupled from the task of long-range language modelling.

We tune the word-level models on validation BPC, and not on segmentation accuracy, so all our models are fully unsupervised with respect to morphological segmentation. Our final hyperparameter settings for SSWM training are similar to those used for SSLM (shown in Table 3.3), except that we use a lower dropout rate (0.2) and higher learning rate (0.005). The maximum subword segments are also longer than for SSLMs, as shown in Table 3.6. The SSWMs train quite slowly, since each word is processed as an individual sequence. Therefore we train our isiXhosa and isiZulu models on 500k-word subcorpora of the language modelling training sets (matching the sizes of the isiNdebele and Siswati datasets). These models converge before 20 epochs, taking 10 hours to train.

Baseline tokenisers Our main goal in applying SSLM to UMS is to determine if its subwords align more closely with morphemes than the subwords of standard tokenisers. Therefore, we apply the BPE and ULM tokenisers from our language modelling baselines (Section 3.6) to the morphological evaluation data, to evaluate them as morphological segmenters. In addition, we turn to a well established family of UMS algorithms, Morfessor [25], for a more competitive baseline. We use the Morfessor Baseline model, which can be trained on a corpus of unsegmented words. Like the ULM tokeniser, Morfessor optimises a unigram language model likelihood. However, to learn morpheme-like subwords, its objective is regularised by a minimum description length (MDL) prior, biasing it towards a compact subword vocabulary and corpus representation. We found that Morfessor performs poorly when trained on our full language modeling corpora, which is consistent with prior findings that it tends to undersegment when applied to large datasets [140]. To evaluate the true potential of Morfessor we view the dataset size as a hyperparameter. We train Morfessor on several subsets of our LM training sets, at different orders of magnitude. The best performance is obtained by training Morfessor on a 10k-word subset, which is the results we present here.

Entropy-based segmenter To explore potentially stronger baselines, we also implement a character-level entropy-based segmenter, based on the work of Mzamo et al. [99]. Their approach consists of training a character-level LM and using the entropy of its probability distribution to predict word segment boundaries. The conditional entropy of x_i in a sequence $\mathbf{x}$ is defined as

$$H(x_i|\mathbf{x}_{<i}) = -\sum_{x \in V} p(x|\mathbf{x}_{<i}) \log p(x|\mathbf{x}_{<i}), \tag{3.9}$$

where V is the character vocabulary (the set of possible values for x_i). This value measures the uncertainty associated with the identity of the next character x_i. The entropy-based segmenter splits words at positions where conditional entropy is high. The motivation behind this is that model uncertainty (entropy) will decrease inside a morpheme and increase at morpheme boundaries, where the next character is harder to predict [39].

Mzamo et al. [99, 100] train n-gram and bi-LSTM LMs for isiXhosa, while Moeng et al. [98] train left-to-right and right-to-left LSTM LMs for isiXhosa, isiZulu, isiNdebele, and Siswati. Their approach consists of training a character-level LM and using the entropy of its probability distribution to predict word segment boundaries. We reuse the character-level LSTM and Transformer LMs trained as baselines in Section 3.6. These LMs were tuned for validation BPC, not segmentation accuracy, since we are applying them as fully *unsupervised* segmenters. We experiment with three entropy-based criteria:

- **Spike:** Predict subword boundary where entropy increases and then decreases.
- **Increase:** Predict subword boundary where entropy increases.
- **Stddev:** Predict subword boundary where entropy exceeds one standard deviation greater than the mean sequence entropy.

3.7.3 Results

Morpheme boundary identification (MBI), the standard measure of morphological segmentation accuracy, casts morphological segmentation as a binary classification problem. For each position in a word, the task is to predict whether a morpheme boundary exists between the characters ($n-1$ binary classification tasks for each word of n characters). Table 3.7 presents the MBI metrics (precision, recall, and F1) for all our methods. We use **boldface** to highlight the top-performing methods respectively among subword segmenters for language modelling and among the baselines developed specifically for UMS. We also **underline** the best performance overall. A few consistent patterns can be observed in the results.

Overall, our word-level subword segmental models (SSWM) are the strongest unsupervised morphological segmenters. In terms of F1 score, SSWM outperforms all baselines (including established methods like Morfessor and entropy-based segmentation) for all four Nguni languages. These performance gains position SSWM as a highly effective approach to UMS. Surprisingly, Morfessor is the weakest among all the models. On the other hand, the entropy-based segmenters achieve competitive results, confirming their potential as unsu-

	xh			zu			nr			ss			*avg*
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1	F1
Subwords for language modelling													
SSLM	30.9	**81.3**	44.8	36.2	**80.7**	**50.0**	37.3	**75.8**	**50.0**	32.5	**61.1**	**42.5**	**46.8**
BPE	40.7	41.8	41.2	45.2	39.0	41.9	38.4	39.8	39.1	32.7	37.8	35.1	39.3
ULM	**45.9**	48.2	**47.1**	**47.0**	42.3	44.5	**41.5**	43.2	42.3	**35.9**	44.0	39.6	43.4
Word-level segmenters													
Morfessor	38.1	25.5	30.5	38.8	31.1	34.5	35.1	32.8	33.9	35.6	31.1	33.2	33.0
SSWM	65.5	50.7	**57.2**	**71.9**	50.7	**59.5**	61.5	54.0	**57.5**	**53.0**	52.9	**53.0**	**56.8**
LSTM entropy-based segmenters													
Spike	51.1	56.3	53.5	53.1	55.0	54.0	46.4	54.8	50.2	41.6	56.6	47.9	51.4
Increase	40.2	63.6	49.2	44.6	63.4	52.4	39.3	**63.0**	48.4	35.5	**65.3**	46.0	49.0
Stddev	**67.6**	40.3	50.5	68.5	39.3	49.9	66.5	41.1	50.8	51.6	42.3	46.5	49.4
Transformer entropy-based segmenters													
Spike	50.4	57.2	53.6	52.1	55.9	53.9	44.4	54.4	48.9	38.0	55.5	45.1	50.4
Increase	42.1	**64.7**	51.0	45.2	**64.6**	53.2	38.4	61.6	47.3	33.2	63.7	43.6	48.8
Stddev	67.0	44.7	53.6	67.7	40.6	50.8	**66.6**	42.2	51.7	51.1	42.2	46.2	50.6

Table 3.7: Morpheme *boundary* identification (MBI) averaged over the evaluation set. The best scores among subword segmenters for language modelling and unsupervised morphological segmenters are **bold**, while the best scores overall are underlined.

pervised segmenters. The strongest entropy-based segmenter is based on inserting subword boundaries on spikes in the entropy of a character-level LSTM.

Our main goal in these experiments is to compare the morphological knowledge encoded by subword segmenters for language modelling. To assess this, we compare the UMS performance of SSLM, BPE, and ULM (the first three rows in Table 3.7). In this respect, we find that the subword boundaries learned by SSLM are closer to morpheme boundaries than those of tokenisers like BPE and ULM. SSLM achieves the highest F1 score among LM subword segmenters for three of the four languages, as well as on average across all the languages. SSLM obtains extremely high recall, but exhibits lower precision than BPE and ULM. This reflects the type of morphological segmentation errors made by SSLM, and reveals something about the nature of its learned subwords. SSLM is correctly identifying a large proportion of the actual morpheme boundaries, but is also often inserting boundaries where it shouldn't. In other words, SSLM has a tendency to over-segment. This could

	xh			zu			nr			ss			*avg*
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1	F1
Subwords for language modelling													
SSLM	20.4	**41.9**	27.5	24.0	**43.7**	**31.0**	**26.4**	**44.6**	**33.2**	19.6	**30.1**	23.7	**28.9**
BPE	25.0	25.4	25.2	25.2	22.9	24.0	21.2	21.7	21.5	22.3	24.4	23.3	23.5
ULM	**33.0**	34.1	**33.5**	**29.0**	27.1	28.0	25.9	26.6	26.2	**25.7**	29.3	**27.4**	28.8
Word-level segmenters													
Morfessor	21.7	17.1	19.1	20.3	17.6	18.8	18.5	17.7	18.1	24.0	22.1	23.0	19.8
SSWM	**44.6**	38.1	**41.1**	**49.4**	**39.8**	**44.1**	**41.4**	**38.0**	**39.6**	**38.3**	**38.3**	**38.3**	**40.8**
LSTM entropy-based segmenters													
Spike	34.7	37.0	35.8	33.6	34.4	34.0	29.1	32.6	30.8	28.9	35.4	31.8	33.1
Increase	27.6	38.1	32.1	29.2	37.5	32.9	25.9	36.4	30.3	24.3	37.0	29.4	31.2
Stddev	41.0	30.4	34.9	39.3	28.2	32.9	38.6	29.0	33.1	34.0	30.4	32.1	33.3
Transformer entropy-based segmenters													
Spike	34.8	37.8	36.3	33.2	34.8	34.0	27.0	31.1	28.9	25.8	33.1	29.0	38.8
Increase	29.7	**40.0**	34.1	29.8	38.5	33.6	24.9	35.0	29.1	22.9	35.9	27.9	31.2
Stddev	43.0	33.9	37.9	39.2	28.9	33.3	38.9	29.7	33.7	33.2	29.7	31.4	34.1

Table 3.8: Morpheme identification (MI) averaged over the evaluation set. The best scores among subword segmenters for language modelling and unsupervised morphological segmenters are **bold**, while the best scores overall are underlined.

be attributed to the low-resource setting in which it is trained. For an LM to utilise long subwords it would have to be exposed to sufficient examples of their usage. This might not be possible with smaller training sets, so the model relies on shorter segments instead.

We also evaluate our models on the task of morpheme identification (MI). MI casts subword segmentation as an information retrieval problem, where a morpheme is correctly identified if it is among the subwords that a word is segmented into. This allows us to measure to what extent SSLM "discovers" morphemes as subword units. Table 3.8 presents MI metrics for all our methods, revealing similar findings as our MBI results. SSWM obtains the best MI precision and F1 scores overall, while SSLM generally obtains better recall. On average, the SSLM subwords match morphemes only slightly better than ULM subwords. So while SSLM learns subword boundaries that align more closely with morphological boundaries (Table 3.7), its tendency to oversegment prevents it from reliably capturing full morphemes.

Language	xh	zu	nr	ss
Morphemes	2.93	2.86	3.03	3.45
SSLM segments	1.43	1.48	1.64	2.24

Table 3.9: Average subword length on UMS test sets.

SSLM might not be a state-of-the-art unsupervised morphological segmenter for the Nguni languages, but it does learn subword boundaries that are linguistically more plausible than those of standard subword tokenisers. This shows that some degree of morphological modelling is being favoured by SSLM, which might contribute to its language modelling performance gains from Section 3.6. In optimising its subword segmentation scheme for language modelling performance, SSLM partially discovers linguistically meaningful subword boundaries.

3.8 Analysis

During the development of SSLM, we experimented with several variations on the subword segmental architecture and performed extensive hyperparameter tuning to find optimal settings for subword segmental training (based on validation set BPC). In this section we report our findings on the effect of architecture and hyperparameters on SSLM performance. We do not analyse these effects for our word-level subword segmental models, since we are primarily interested in investigating which components contribute to the success of subword segmental modelling for language modelling.

Kawakami et al. [70] found two components to be crucial to the success of their segmental LM for Chinese word discovery: the subword lexicon and expected length regularisation. The former stores frequent subwords and the latter introduces a regularisation term to the training objective to encourage shorter segments. We initially implemented both of these components for SSLM. However, contrary to Kawakami et al. [70], we did not find length regularisation to be useful. This is because our training sets are much larger than those used in their ablation studies (they use the Brent corpus of 27k words). When a segmental model

isiNdebele sentence segmentation	
Sentence	Sibuye sithokoze khulu kwamanikelela emphakathini weentjhabatjhaba ngesekelo labo elinganakuzaza emzabalazweni wethu.
Morphemes	Si-buy-e si-thokoz-e khulu k-w-amanik-elel-a e-m-phakath-ini weentjhabatjhaba nge-sekelo labo eli-nga-nakuzaza e-mzabalazw-eni w-ethu.
SSLM	Si-buy-e s-i-t-h-oko-z-e k-hulu kwam-a-nikele-l-a e-m-phakath-i-n-i ween-tjhaba-tjhab-a n-g-e-sekelo l-a-b-o e-l-i-ngana-kuz-az-a e-mz-abal-az-w-e-n-i w-e-thu.
BPE	Si-bu-ye si-tho-ko-ze khulu kwa-m-ani-k-elela em-phaka-thini ween-tjhaba-tjhaba nge-se-k-elo la-bo eli-ng-ana-ku-za-za em-za-bala-z-weni we-thu.
ULM	Si-bu-ye si-tho-ko-ze khulu kwama-nikele-la emphakathin-i w-eentjhabatjhab-a nge-se-ke-lo la-bo e-lingana-ku-za-za em-za-ba-la-zwe-ni we-thu.

Table 3.10: Subword segmentations compared to the annotated morphological segmentation of an isiNbedele sentence. Correctly identified morphemes are highlighted.

is trained on a small dataset, it overfits by modelling long segments with the lexicon. Length regularisation overcomes this by biasing the model towards shorter segments. This is not a problem on larger datasets like ours, because the lexicon cannot possibly include all the long segments that occur in the corpus. Even without length regularisation, the model learns shorter segments. In fact, our model tends to over-segment rather than under-segment, as shown in our UMS experiments. Table 3.9 confirms this, showing that SSLM subwords are on average much shorter than morphemes. It is also evident in Tables 3.10 & 3.11, which contain examples of segmented Nguni-language sentences. For SSLM, we use the Viterbi algorithm to extract the likelihood-maximising subword segmentations. SSLM often over-segments, but sometimes its segmentations are more morphologically accurate because of its tendency towards shorter segments. In the examples, BPE and ULM fail to identify most of the 1-character morphemes, while the SSLM identifies several.

(a) isiXhosa sentence segmentation

Sentence	Siphinda kwakhona umbulelo wethu osuka emazantsi entliziyo kwabezizwe ngezizwe ngenkxaso yabo engagungqiyo ekuxhaseni umzabalazo wethu.
Morphemes	Si-phind-a kwa-khona u-m-bulelo w-ethu o-suk-a emazantsi e-n-tliziyo kwabezizwe nge-zi-zwe ngenkxaso y-abo engagungqiyo e-ku-xhas-eni u-m-zabalazo w-ethu.
SSLM	S-i-phin-d-a kwak-hon-a u-m-bule-l-o w-e-thu osuk-a e-m-a-zant-s-i e-n-tliziy-o k-w-a-b-e-z-i-zw-e ngezi-zw-e n-g-e-nkxa-s-o y-a-b-o e-ngag-ungqi-yo e-k-u-xhas-e-n-i u-m-zab-alaz-o w-e-thu.
BPE	Si-phin-da kwa-khona um-bu-lelo we-thu o-su-ka ema-zantsi ent-li-zi-yo kwa-b-ezi-zwe ngezi-zwe ngen-kxaso ya-bo enga-gu-ng-qi-yo eku-xha-s-eni um-za-bala-zo we-thu.
ULM	Si-phi-nda kwa-khona u-mbu-lelo we-thu o-suka e-ma-za-nts-i e-nt-li-zi-yo kwa-be-zi-z-we nge-zi-z-we nge-nkxaso ya-bo e-nga-gu-ngq-i-yo e-ku-xh-a-se-ni um-za-ba-la-zo we-thu

(b) isiZulu sentence segmentation

Sentence	Siyaphinda sibonga siyanconcoza emphakathini womhlaba ngokuseseka kwawo emzabalazweni wethu.
Morphemes	Si-ya-phind-a si-bong-a si-ya-nconcoz-a e-m-phakath-ini wo-m-hlaba n-gokusesek-a kwa-wo e-mzabalazw-eni w-ethu.
SSLM	S-i-yaph-inda s-i- bong-a siya-nco-nco-z-a e-mphak-a-t-h-i-n-i w-o-m-hlaba n-g-o-k-u-s-eseka k-w-a-w-o emz-abala-z-w-e-n-i wethu.
BPE	Si-ya-phi-nda si-bo-nga si-ya-n-co-n-co-za em-phakathi-ni wo-m-hlaba ngoku-se-se-ka kwa-wo em-za-bala-zweni we-thu.
ULM	Si-ya-phi-nda si-bo-nga si-ya-n-co-n-co-za emphakathini wo-mhlaba ngoku-se-se-ka kwa-wo em-za-ba-la-zwe-ni we-thu.

(c) Siswati sentence segmentation

Sentence	Sendlulisa kubonga kwetfu kummango wemave emhlaba ngekwesekela umzabalazo wetfu ngendlela lengenakunyakatiswa.
Morphemes	S-endlulis-a ku-bong-a kwetfu ku-mmango wemave emhlaba ngekwesekela u-mzabalazo wetfu nge-n-dlela le-n-genakunyakatiswa.
SSLM	S-e-n-dlulis-a k-u-bong-a kw-e-tfu k-u-m-mango w-e-mave emhlaba n-g-e-k-w-e-sekel-a u-m-zaba-laz-o w-e-tfu ngendle-l-a l-e-ngena-ku-n-yaka-t-isw-a
BPE	S-en-dlu-lisa kubo-nga kwetfu kum-ma-ngo wema-ve em-hlaba ngekwe-se-kela um-za-bala-zo we-tfu ngendlela le-ngen-aku-nya-kati-swa.
ULM	Se-ndlu-lisa kubo-nga kwe-tfu ku-m-ma-ngo we-mave e-mhlaba ngekwe-se-kela um-za-ba-la-z-o we-tfu ngendlela le-ngena-ku-nya-kati-swa.

Table 3.11: Subword segmentations compared to the annotated morphological segmentations of Nguni-language sentences. Correctly identified morphemes are highlighted.

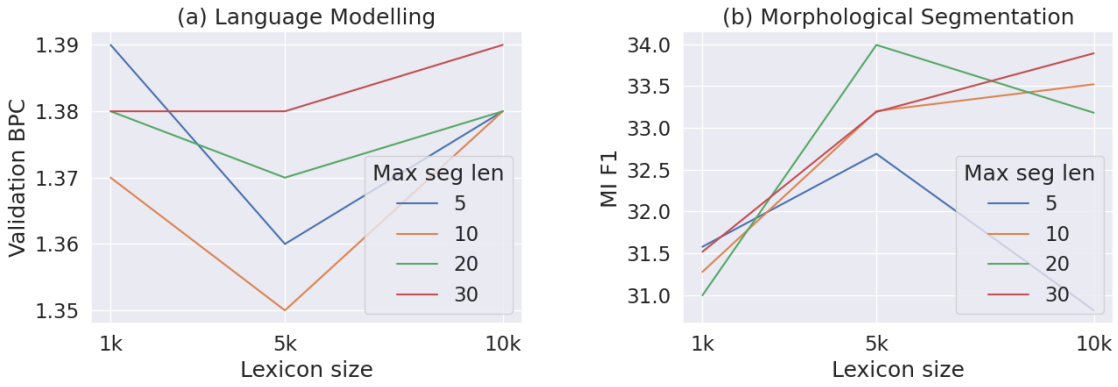

Figure 3.2: Comparing isiNdebele SSLM performance on language modelling and unsupervised morphological segmentation across varied lexicon sizes and maximum segment lengths.

As described in Section 3.4, our model is equipped with a subword lexicon consisting of the V most frequency subwords up to length L, where both V and L are prespecified hyperparameters. The lexicon biases the SSLM to high-frequency subwords. This proves essential for SSLM performance, consistently leading to improved validation BPC. The average lexicon coefficient ($1 - g_t$ in equation 3.5) of the isiNdebele SSLM on the LM test set was 0.27, indicating a reliance on the lexicon for subword generation. We analyse two lexicon-related hyperparameters: lexicon size and maximum segment length. The size of the lexicon is pre-specified, similar to how the BPE vocabulary size is pre-specified. The maximum segment length determines the length (number of characters) of the longest subword segments considered by SSLM during training. We tuned both to optimise validation BPC. Figure 3.2 compares the performance of isiNdebele SSLMs across varying lexicon sizes and maximum segment lengths. Figure 3.2 (a) plots intrinsic LM performance as measured by BPC. Smaller lexicon sizes improve LM performance up to a point, with 5k subwords being optimal. A maximum segment length of 10 characters achieves optimal performance across all lexicon sizes, striking a balance between memorising long segments where possible, and otherwise relying on short subwords.

Figure 3.2 (b) plots UMS performance, where the picture is less clear-cut. Since our model is an *unsupervised* morphological segmenter, we only considered LM performance (validation BPC) when tuning and selecting our final models. Figure 3.2 shows that optimal

LM performance does not necessarily imply optimal UMS performance. Longer maximum segment lengths tend to improve UMS performance. Biasing the model towards longer segments reduces the over-segmentation problem, but relinquishes some LM performance. However, the graphs demonstrate that there is at least some correlation between LM and UMS performance, and the model selected for LM performance is not far off optimal UMS accuracy.

3.9 Limitations

Our experiments are limited to the Nguni languages, which are highly related and morphologically similar. Our findings might not hold for languages with different types of morphological complexity (e.g. fusional languages, where segmentation is difficult because morphemes are fused together). SSLM achieves consistently good LM performance across all four Nguni languages, but required validation-based tuning of the lexicon size and maximum segment length, separately for each language. The optimal values for these hyperparameter vary across languages and would have to be tuned from scratch for new languages.

Our subword segmental approach is able to improve over all baselines as a morphological segmenter, but only if we train it as a word-level sequence model. SSLM outperforms tokenisers like BPE and ULM, but performs worse than entropy-based segmenters. The disparity in UMS performance between our word-level models and SSLM shows that the long-range linguistic modelling required of SSLM impedes morphologically accurate segmentation. This is also evident in the fact that SSLM sometimes reverts to character-level modelling to optimise its language modelling training objective.

We only evaluate our segmentations with automatic evaluation metrics, which provides a rigid, morpheme-based perspective on the segmentation quality. It would be ideal to include human assessments of the linguistic plausibility of segmentations.

3.10 Ethical Considerations

We release language modelling datasets for four Nguni languages, consisting of unannotated text corpora split into train/validation/test sets. Our datasets are compilations of existing, publicly available datasets (listed in Table 3.2). As outlined in Section 3.5.1, we performed preprocessing and data cleaning to ensure that the datasets were of reasonable quality. Nevertheless, the source datasets were originally scraped from the web, so we acknowledge that we do not avoid all the pitfalls of large scale data collection for low-resource languages (most notably, the presence of text in other languages). Furthermore, most of the data is sourced from South African government publications, so they are domain-specific to some extent. The texts cover diverse topics, but generally fall within the categories and style expected of government publications.

3.11 Conclusion

In this chapter we proposed SSLM, which unifies autoregressive language modelling and subword segmentation in an end-to-end trainable model. We train SSLMs for each of the four Nguni languages of South Africa. We compile LM datasets for these languages from publicly available corpora and release our train/validation/test sets. On intrinsic evaluation (test set perplexity-based metrics) averaged across the Nguni languages, SSLM outperforms neural LMs trained with BPE and ULM subwords, as well as character-level LMs. On the task of unsupervised morphological segmentation, which determines to what extent subwords align with actual morphemes, SSLM again outperforms subword tokenisation on average across the four languages.

In addition to these LMs, we train a second set of subword segmental models that train on single words in isolation (without having to model context for long-range language modelling). Our word-level models outperform existing methods for unsupervised morphological segmentation, including well known unsupervised segmenters like Morfessor, by large margins across all Nguni languages. Finally, we discuss some of our findings in developing the SSLM. We highlight the importance of including a subword lexicon in our architecture, and

analyse how lexicon-related hyperparameters affect performance. In summary, this chapter presents the following contributions:

1. We propose the SSLM, which unifies subword segmentation and language modelling in a single end-to-end neural architecture.

2. We compile and release curated LM datasets for the four Nguni languages of South Africa (isiXhosa, isiZulu, isiNdebele, and Siswati).

3. We train SSLM models for the four Nguni languages and find that it outperforms tokenisation-based LMs on perplexity-based test set evaluation.

4. We evaluate SSLM on the task of unsupervised morphological segmentation, demonstrating that it learns Nguni subwords that are more aligned with morphemes than BPE and ULM subword tokens.

5. We train word-level SSLMs, which prove highly effective as unsupervised morphological segmenters, comfortably outperforming established unsupervised segmenters.

6. We present an analysis of how hyperparameters specific to subword segmenta modelling affect LM performance.

SSLM improves intrinsic LM performance for the Nguni languages, and learns subword boundaries that are closer to morpheme boundaries than subword tokenisers. As the first successfully implemented subword segmental model, SSLM validates our approach of learning subword segmentation during training and offers a promising alternative to tokenisation-based models. Due to limited pretraining and downstream task datasets for Nguni-language text generation, our experiments in this chapter were limited to intrinsic evaluation metrics. However, the techniques underlying SSLM can now be adapted to other neural architectures and tasks. In the following chapter we do exactly that, scaling subword segmental modelling to larger models and applying them to a task on which we can evaluate text generation.

4

Subword Segmental Machine Translation

In this chapter we propose our second subword segmental model, for the task of machine translation (MT). Subword segmental machine translation (SSMT) extends our techniques to the Transformer-based encoder-decoder architecture. While being trained to translate from source to target language, SSMT jointly learns subword segmentation for the target language that optimises translation performance. MT is the only text generation task for which the Nguni languages have large-scale web-scraped datasets available, so SSMT is the first of our models we can evaluate for text generation. Generating text with subword segmental models proves non-trivial, since standard beam search cannot be applied to the subword segmental architecture. To address this we propose dynamic decoding, an algorithm for subword segmental text generation. We train bilingual SSMT models for English to six target languages, including three Nguni languages. SSMT matches or outperforms the best tokenisation-based NMT baselines on translation to isiXhosa, isiZulu, and Finnish (another agglutinative language). SSMT achieves its largest gains (+2.5 chrF) on the lowest resourced translation direction of English to Siswati. As in the case of SSLM, we show that SSMT learns subwords that are closer to morphemes than standard subword tokens. SSMT also exhibits more robust performance on an isiZulu test set constructed to test morphological compositional generalisation (the ability to generalise to novel combinations of known morphemes), demonstrating that learning subword *for* translation allows SSMT to more effectively compose the meaning of words from their constituent morphemes.

Train		
I do understand.	→	Ndi-ya-qonda.
I am tired.	→	Ndi-diniwe.
Where are you from?	→	U-vela phi?
Are you busy?	→	Ingaba u-xakekile?
Test		
Do you understand?	→	U-ya-qonda?
I am busy.	→	Ndi-xakekile.

Table 4.1: Parallel English-isiXhosa sentences with morphologically segmented isiXhosa words. The train/test split shows why its critical to accurately model morphemes and morphological compositional generalisation i.e. novel combinations of known morphemes.

4.1 Motivation

The success of neural machine translation (NMT) is closely tied to subword segmentation. In fact, tokenisers like BPE [135] and ULM [81], now integral to most NLP tasks, were first introduced for MT. Practitioners typically run subword tokenisation on the combined parallel corpus during preprocessing, producing a single bilingual/multilingual tokeniser that is applied to all languages. This approach requires models to learn robust mappings between source-target tokens, which is particularly challenging when training data is scarce and the languages involved are morphologically rich with complex subword structures.

As a result, subword segmentation remains an active area of research in MT. For low-resource languages, where handling rare words is crucial, standard tokenisers are outperformed by unconventional approaches like character-level models [37], quasi character-level models with very small subword vocabularies [16], and subword segmentation based on self-supervised learning [141]. For morphologically rich languages, standard tokenisers are outperformed by methods that incorporate linguistic information [60, 139], highlighting the shortcomings of data-driven segmentation. The same holds for languages that are both morphologically rich *and* under-resourced, where the importance of subword segmentation is further exacerbated [106, 132, 131].

As in the case of language modelling, the underperformance of subword tokenisation in these settings can be partially attributed to the fact that it separates subword segmentation

from model training. BPE and ULM are applied before training starts, so models are reliant on fixed subword segmentations. This is not ideal, since these tokenisers are not guaranteed to produce subword units that are optimal for MT performance. MT is the task of mapping source sentences to target sentences, both of which are represented as a sequence of subword units. The difficulty of MT as a task partially depends on properties of source and target subword units, such as the degree to which they are aligned between languages to allow to for source-target mapping. Ideally, we would want to learn subword units *for* MT, which can be achieved by jointly learning translation and subword segmentation.

There have been two efforts towards optimising subword segmentation (respectively of source and target languages) for MT performance. Kreutzer and Sokolov [80] learn source sentence segmentation during MT training by equipping their encoder with a halting unit [51] to dynamically determine how many inputs a source segment should consist of. Their models end up preferring character-level segmentation of source sentences, and fail to improve MT performance over BPE and character NMT models. He et al. [56] propose dynamic programming encoding (DPE), which trains an NMT model that marginalises over target sentence segmentations. After training they apply their model as a subword segmenter by computing the maximising segmentations, and train new NMT models from scratch on the resulting segmented corpora. DPE is still a preprocessing step (a separate vanilla NMT model is trained on a corpus segmented by DPE), but since its segmentations are trained on MT, they are at least connected to the task at hand. In this chapter we go one step further by fully unifying NMT and subword segmentation.

We propose subword segmental machine translation (SSMT), an end-to-end NMT model that learns subword segmentation during training and can be used directly for inference. It is trained with a dynamic programming algorithm that allows the model to learn a subword segmentation scheme that optimises its MT training objective. The architecture is a Transformer-based encoder-decoder adaptation of the subword segmental language model (SSLM) of Chapter 3 that jointly learns MT and target-side segmentation. We also propose *dynamic decoding*, a decoding algorithm for subword segmental models that dynamically adapts subword segmentations as it generates translations. It extends subword segmental

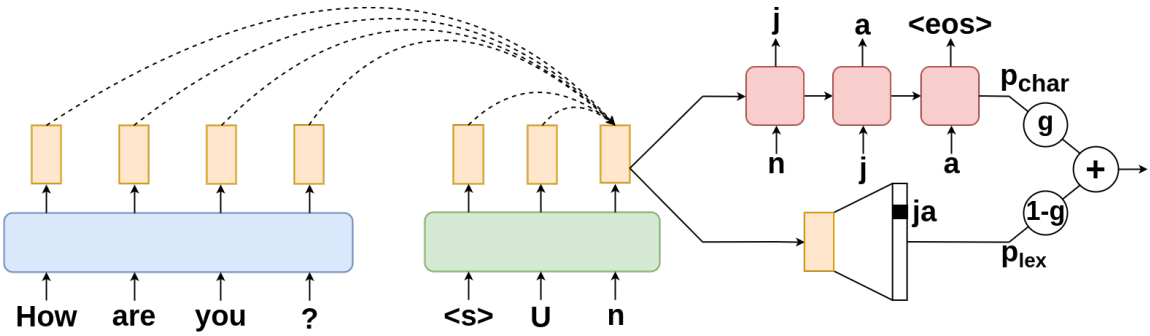

Figure 4.1: SSMT translates "How are you?" to the isiZulu "Unjani?", computing the probability for subword "ja". A Transformer encoder-decoder encodes the BPE-segmented source sentence and character-level target sentence. A mixture between a character decoder and lexicon model (Equation 4.3) produces the next subword probability.

modelling to inference time, enabling the model to generate translations based on its optimised subword segmentations. The fact that SSMT is used directly to generate translations sets it apart from previous MT-oriented segmenters like DPE. SSMT is not a preprocessing step in any sense – it is single model for translation, subword segmentation, and generation.

4.2 Neural Architecture

SSMT is a Transformer-based encoder-decoder (Figure 4.1). The encoder is that of a vanilla Transformer NMT model. Source language sentences are pre-segmented with BPE. The decoder adapts the subword segmental architecture of Chapter 3 to be Transformer-based (as opposed to our LSTM-based SSLM) and conditioned on the source sentence.[1] During training, SSMT considers all possible subword segmentations of the target sentence and learns which of these optimise its translation training objective.

Given a source sentence of BPE tokens $\mathbf{x} = x_1, x_2, ..., x_{|\mathbf{x}|}$, SSMT generates the target sentence of characters $\mathbf{y} = y_1, y_2, ..., y_{|\mathbf{y}|}$ as a sequence of subwords $\mathbf{s} = s_1, s_2, ..., s_{|\mathbf{s}|}$. We introduce a conditional semi-Markov assumption, whereby each subword probability is

[1] It would be possible to have a subword segmental encoder and decoder. The main goal of this thesis is to learn subword segmentation to optimise text generation, so we chose to only adapt the decoder, which is the text generation component in NMT. SSMT learns subword segmentation that optimises MT performance, conditioned on the BPE segmentation scheme of its source language.

computed as

$$p(s_i|\mathbf{s}_{<i},\mathbf{x}) \approx p(s_i|\pi(\mathbf{s}_{<i}),\mathbf{x}) \tag{4.1}$$

$$= p(s_i|\mathbf{y}_{<j},\mathbf{x}), \tag{4.2}$$

where $\pi(\mathbf{s}_{<i})$ is a concatenation operator that converts the sequence $\mathbf{s}_{<i}$ into the raw unsegmented characters $\mathbf{y}_{<j}$ preceding subword $\mathbf{s}_i$ in the target sentence. We condition on the unsegmented history for computational efficieny, to avoid having to incorporate all possible preceding subword segmentations. We condition on the source and target sentence history by passing the final output embedding of the Transformer decoder (green in Figure 4.1) to two subnetworks, an LSTM for character-by-character generation (red in Figure 4.1) and an MLP for subword generation (the linear layer in Figure 4.1), that compute subword probabilities.

The final subword probability of Equation 4.2 is based on a mixture of these subnetworks,

$$p(s_i|\mathbf{y}_{<j},\mathbf{x}) = g_j p_{\text{char}}(s_i|\mathbf{y}_{<j},\mathbf{x}) + (1-g_j)p_{\text{lex}}(s_i|\mathbf{y}_{<j},\mathbf{x}), \tag{4.3}$$

which combines probabilities from a character LSTM (p_{char}) and a 1-layer MLP with a softmax activation, which outputs probability (p_{lex}) if s_i is in the lexicon (otherwise $p_{\text{lex}} = 0$). The lexicon contains the V most frequent character sequences (n-grams) up to some maximum segment length in the training corpus (V is a prespecified vocabulary size). The lexicon models frequent subwords (e.g. common morphemes), while the character decoder models rare subwords and previously unseen words (e.g. it can copy names from source to target sentences). The mixture coefficient g (computed by a fully connected linear layer) allows SSMT to learn, based on context, when the next subword is likely to be in the lexicon and when it should rely on character-level generation.

4.3 Dynamic Programming Algorithm for Training

We use this architecture to train a model that jointly learns translation and target-side subword segmentation. The subword segmentation of a target sentence is treated as a latent variable

and marginalised over to compute the probability

$$p(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{s}:\pi(\mathbf{s})=\mathbf{y}} p(\mathbf{s}|\mathbf{x}), \tag{4.4}$$

where the probability of a specific subword segmentation $\mathbf{s}$ is computed with the chain rule as a product of its individual subword probabilities (each computed as Equation 4.3).

We can compute this marginal efficiently by adapting our dynamic programming training algorithm from Chapter 3 to incorporate conditional probabilities. At each character position k in the raw target sentence $\mathbf{y}$ the forward probability is,

$$\alpha_k = \sum_{j=f(\mathbf{y},k)}^{k} \alpha_{j-1} p(s = \mathbf{y}_{j:k} | \mathbf{y}_{<j}, \mathbf{x}), \tag{4.5}$$

with $\alpha_0 = 1$. The function $f(\mathbf{y},k)$ outputs the starting index of the longest possible subword ending at character k. This will either be $k-m$, where m is the maximum segment length (a pre-specified hyperparameter) or it will be the starting index of the current word (if character $k-m$ precedes the start of the current word).

This last constraint is critical, since it limits the model to learn segmentation of words into subwords. The function $f(\mathbf{y},k)$ ensures that our model cannot consider segments that cross word boundaries; the only valid segments are those within words. SSMT model is optimising subword segmentation, as opposed to the sequence segmentation of raw text. Characters that separate words (e.g. spaces and punctuation) are treated as 1-character segments. In this way we also implicitly model the beginning and end of words, since these are the boundaries of valid segments.

4.4 Dynamic Decoding

The standard method for generating translations in tokenisation-based NMT is to run beam search over the subword vocabulary. However, the SSMT mixture model (Equation 4.3) has two vocabularies, a character vocabulary and a subword lexicon. While beam search can

be applied separately to either one, to approximate finding the highest scoring translation according to SSMT, subword prediction should be based on the full mixture distribution. To achieve this our decoding algorithm has to incorporate information from both the character and lexicon models.

During training, SSMT considers all possible segmentations of the target sentence using a dynamic programming algorithm. We would like to consider different segmentations during generation as well, instead of being limited to the subword boundaries dictated by greedy prediction. This requires retaining part of the dynamic program during decoding, a technique previously employed for modelling latent alignments between multi-word segments [171] and latent dependencies [13]. Doing so would enable SSMT to dynamically adjust the preferred segmentation as new subwords are generated.

In this section we outline *dynamic decoding*, an algorithm that (1) incorporates both the character and lexicon models and (2) dynamically adjusts subword segmentation during generation. Dynamic decoding modifies the next-subword probability computation and the decoding search algorithm, in order to consider different possible subword segmentations during generation.

4.4.1 Next Character Prediction

Dynamic decoding generates one character at a time and computes next-character probabilities with the full mixture model. To explicitly model subword boundary decisions, we distinguish between characters that represent subword boundaries and subword continuations. For each generated character, the decoding algorithm additionally specifies whether the character ends a subword (it is the last character in the subword) or whether it continues a subword (more characters will follow in the subword). The mixture model's next-character probability calculation is different, depending on whether we compute the probability of the next character ending the current subword (denoted **end**) or continuing the current subword (denoted **con**).

Similarly, at each character generation step we have to consider whether the *preceding* character ends or continues a subword. If it ends a subword, then the next character starts a

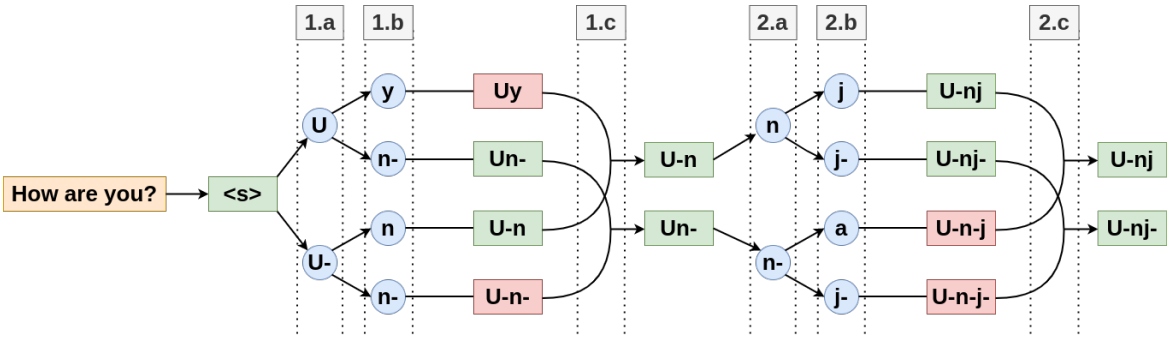

Figure 4.2: Dynamic decoding for the first 2 characters of a translation ("-" are subword boundaries). Step (a) produces candidate characters that continue and end the subword. Step (b) peaks one character ahead. Step (c) finalises the segmentation decision. Green sequences are chosen ahead of red ones based on higher sequence probabilities.

new subword. If the preceding character continues a subword, then the next character is the latest addition to the current subword. These considerations also affect how the next-character probability is computed.

Given this setup, we have 4 possible cases for next-character generation (corresponding to the four nodes in Steps 1.b and 2.b in Figure 4.2):

1. **con-end** – the preceding character continues a subword that the next character ends,
2. **end-con** – the preceding character ends a subword and the next character starts a new one,
3. **end-end** – both preceding and next characters end subwords (the latest subword is a 1-character subword),
4. **con-con** – both preceding and next characters continue the same subword.

Each case uses the mixture model differently to compute next-character probabilities. We now present the formulas for all 4 cases, defining the probabilities used in our dynamic decoding algorithm ($p_{\text{con-end}}, p_{\text{end-con}}, p_{\text{end-end}}, p_{\text{con-con}}$), which is outlined in Algorithm 1.

We consider the simplest case first. Given that the previously generated character at position $j-1$ concludes a subword, the probability of the next subword being a single character y is

$$p_{\text{end-end}}(y|\mathbf{y}_{<j},\mathbf{x}) = g_j p_{\text{char}}(y,\text{<eos>}\,|\,\mathbf{y}_{<j},\mathbf{x}) + (1-g_j)p_{\text{lex}}(y|\mathbf{y}_{<j},\mathbf{x}), \tag{4.6}$$

Input: $\mathbf{x}$ is a source sentence of BPE tokens
Output: $\mathbf{y}^*$ is the generated translation, a character sequence concluding with <eot> (end-of-translation)
Notation: C is a character vocabulary
$\mathbf{y}_{\text{end}}$: partial translation, last char ends subword
$\mathbf{y}_{\text{con}}$: partial translation, last char continues subword

$$y_{\text{con}} = \arg\max_{y \in C} p_{\text{end-con}}(y|\mathbf{x}), \mathbf{y}_{\text{con}} = [y_{\text{con}}]$$
$$y_{\text{end}} = \arg\max_{y \in C} p_{\text{end-end}}(y|\mathbf{x}), \mathbf{y}_{\text{end}} = [y_{\text{end}}]$$

while $\mathbf{y}_{\text{end}}[-1] \neq$ <eot> **do**

$$y_{\text{con-con}} = \arg\max_{y \in C} p_{\text{con-con}}(y|\mathbf{y}_{\text{con}}, \mathbf{x})$$
$$y_{\text{end-con}} = \arg\max_{y \in C} p_{\text{end-con}}(y|\mathbf{y}_{\text{end}}, \mathbf{x})$$
$$\mathbf{y}_{\text{con}} = \arg\max_{\mathbf{y} \in \{[\mathbf{y}_{\text{con}}, y_{\text{con-con}}], [\mathbf{y}_{\text{end}}, y_{\text{end-con}}]\}} p(\mathbf{y})$$
$$y_{\text{con-end}} = \arg\max_{y \in C} p_{\text{con-end}}(y|\mathbf{y}_{\text{con}}, \mathbf{x})$$
$$y_{\text{end-end}} = \arg\max_{y \in C} p_{\text{end-end}}(y|\mathbf{y}_{\text{end}}, \mathbf{x})$$
$$\mathbf{y}_{\text{end}} = \arg\max_{\mathbf{y} \in \{[\mathbf{y}_{\text{con}}, y_{\text{con-end}}], [\mathbf{y}_{\text{end}}, y_{\text{end-end}}]\}} p(\mathbf{y})$$

end
$\mathbf{y}^* = \mathbf{y}_{\text{end}}$
return $\mathbf{y}^*$

Algorithm 1: Dynamic decoding is an approximate search algorithm for a probability-maximising translation. During training, we marginalise over all possible segmentations. During decoding, we seek to generate the single, optimal segmentation of a translation.

where <eos> is a special end-of-subword token. We compute this for all y in the character vocabulary and return the top candidates for the next character. We modify this for the case where character $j-1$ does not conclude a subword, but character j still does. Then character j constitutes the last character in a subword that started at an earlier character. The probability of next character is then

$$p_{\text{con-end}}(y|\mathbf{y}_{<j}, \mathbf{x}) = g_j p_{\text{char}}(y, \text{<eos>} \,|\, \mathbf{y}_{k:j-1}, \mathbf{y}_{<k}, \mathbf{x}) + (1 - g_j) p_{\text{lex}}(y|\mathbf{y}_{k:j-1}, \mathbf{y}_{<k}, \mathbf{x}), \quad (4.7)$$

where k is the starting position of the current subword (concluding at j) and $\mathbf{y}_{k:j-1}$ are the characters generated so far in the current subword.

These cases still only give us candidates for when the next character concludes a subword. We can modify Equation 4.6 to compute the probability of the next character starting and

continuing a subword as

$$p_{\text{end-con}}(y|\mathbf{y}_{<j},\mathbf{x}) = g_j p_{\text{char}}(y|\mathbf{y}_{<j},\mathbf{x}) + (1-g_j) \sum_{\mathbf{s}:s_1=y,\mathbf{s}\neq y} p_{\text{lex}}(\mathbf{s}|\mathbf{y}_{<j},\mathbf{x}). \tag{4.8}$$

where the first mixture component is simply the probability of the next character under the character-level model (without the <eos> token, since the subword is not concluded). The second component marginalises over all subwords in the lexicon that begin with y, accounting for every possible way the lexicon could generate character y as the start of a new subword. It excludes the 1-character subword y ($\mathbf{s} \neq y$), since this constitutes a subword ending with character j (covered by Equation 4.6). Like Equation 4.6, this covers the case in which the previous character concludes a subword. Mirroring the extension of Equation 4.6 to Equation 4.7, we can generalise Equation 4.8 to the case where character j continues a subword started at any given previous character, resulting in the following final probability calculation:

$$p_{\text{con-con}}(y|\mathbf{y}_{<j},\mathbf{x}) = g_j p_{\text{char}}(y|\,\mathbf{y}_{k:j-1},\mathbf{y}_{<k},\mathbf{x}) + (1-g_j) \sum_{\mathbf{s}:s_1=y,\mathbf{s}\neq y} p_{\text{lex}}(\mathbf{s}|\,\mathbf{y}_{k:j-1},\mathbf{y}_{<k},\mathbf{x}). \tag{4.9}$$

4.4.2 Dynamic Segmentation

One could use these next-character probabilities to greedily generate translations one character at a time, inserting subword boundaries when $p_{\text{con-end}} > p_{\text{con-con}}$ or $p_{\text{end-end}} > p_{\text{end-con}}$. When a subword is concluded during generation, that subword boundary would be final. However, this would amount to a greedy search over the space of possible subword segmentations, which might be sub-optimal given characters that are subsequently generated. Ideally the decoding algorithm should make the final segmentation decision based on characters to the left and right of a potential subword boundary. This would enable the decoding algorithm to reconsider past segmentation decisions if they turn out to be sub-optimal, given characters that are generated after them. To achieve this we design a decoding algorithm that retains part of the dynamic program during generation (see Algorithm 1).

For simplicity we explain dynamic decoding for a beam size of 1, but it is straightfoward to extend the algorithm to multiple beams. Figure 4.2 demonstrates the generation of the first few characters of a translation. The key is to hold out on finalising segmentations until subsequent characters have been generated. We compute candidates for the next character, but do so separately for candidates that continue the current subword and those that end the current subword (step (a) in Figure 4.2). The segmentation decision is postponed until after the next character has been generated. We now essentially have two "potential" beams – one for continuing the current subword and another for ending it. For each of these potential beams, we repeat the previous step: we compute candidates for the next character, keeping separate the candidates that continue and end the subword (step (b) in Figure 4.2).

Now we reconsider past segmentations. We compare sequence probabilities across the two potential beams of the character generated one step back (comparisons are visualised by arcs under step (c)). We select the best potential beam that continues the current subword and the best potential beam that ends the current subword. We then repeat the process on these new potential beams. Essentially we are retrospectively deciding whether the previous character should end a subword. Since we have postponed the decision, we are able to consider how it would affect the generation of the next character. For example, in step (2.c) of Figure 4.2, the subword boundary after character "n" is reconsidered and discarded, given that it leads to lower probability sequences when we generate one character ahead.

Figure 4.2 visualises dynamic decoding for a single beam, although at any point the algorithm is storing at least two character sequences – the top beam concluding with a subword boundary and the top beam concluding with an incomplete subword. To extend dynamic decoding to n beams, each character proposal step in dynamic decoding should suggest the top n characters, instead of just the top character. The $\arg\max$ operators in Algorithm 1 should return the characters corresponding to the top n probabilities. Dynamic decoding with multiple beams always stores n beams that conclude with continuing subwords and n beams that conclude with ending subwords, so effectively the algorithm is storing $2n$ candidate character sequences. During generation, we normalise beam probabilities by sequence length (number of subword segments) to prevent bias towards shorter sequences.

During training, we incorporate all possible subword segmentations of a target sentence, by marginalising with a dynamic programming algorithm. During decoding, we retain part of this dynamic program to evaluate all possible segmentations of the two most recently generated characters. In this way, our decoding algorithm dynamically adjusts segmentations during generation, allowing the model to insert subword boundaries that better approximate maximising translation probabilities.

4.5 Experimental Setup

We implement SSMT as a sequence-to-sequence model in the open-source fairseq library[2] and publicly release our code and trained models.[3] In all our experiments, we compare SSMT to NMT models based on three tokenisers – BPE, ULM, and DPE. We now present the languages and datasets used in our experiments, after which we detail hyperparameter tuning and our training procedure.

4.5.1 Data

We train bilingual MT models from English to six languages. As shown in Table 4.2, the chosen languages allow us to compare how effective SSMT is across three different morphological typologies – agglutinating conjunctive, agglutinating disjunctive, and analytic. Our primary focus is the Nguni languages, which are agglutinating conjunctive, so this morphological grouping makes up the majority of our translation directions. We include three Nguni languages (isiNdebele is not covered by our training dataset), as well as Finnish, to test SSMT on a morphologically complex language unrelated to the Nguni languages. We include Setswana (agglutinating disjunctive) and Afrikaans (analytic), both of which have short word forms compared to the Nguni languages, to test SSMT on morphologically simple languages. For English to Finnish we train on Europarl[4], while for the other directions we

[2]https://github.com/facebookresearch/fairseq
[3]https://github.com/francois-meyer/ssmt
[4]https://www.statmt.org/europarl/

Language	Morphology	Orthography	Family	# Sentences	Dataset
isiXhosa	agglutinative	conjunctive	Nguni	8.7mil	WMT22
isiZulu			Nguni	3.9mil	WMT22
Siswati			Nguni	165k	WMT22
Finnish			Uralic	1.6mil	Europarl
Setswana	agglutinative	disjunctive	Sotho–Setswana	5.9mil	WMT22
Afrikaans	analytic	disjunctive	Germanic	1.6mil	WMT22

Table 4.2: Linguistic typology and training data sizes for the target languages used in our experiments (source language is always English).

train on WMT22_African.[5] The parallel corpora sizes are listed in Table 4.2. Our language selection covers different levels of data availability, with English → Siswati being by far the least resourced instance with only 165k parallel sentences.

4.5.2 Hyperparameters

For all our models we use the settings of the fairseq transformer-base architecture[6] (6 encoder layers, 6 decoder layers). We use the FLORES [148, 49] development set as our validation set (997 multi-parallel sentences) to optimise hyperparameters relevant to subword modelling. We extensively tune the vocabulary sizes of our models on both English → isiXhosa and English → isiZulu (including the use of separate source and target vocabularies). Validation performance peaks for both translation directions at a shared vocabulary of 10k subwords for the tokenisation-based baselines. For SSMT it peaks at 5k BPE subwords for the source language and 5k subwords in the target language lexicon. We apply these vocabulary settings to the remaining languages, excluding Siswati, which we tune separately. Since models are more sensitive to hyperparameter settings in data scarce settings [7], we perform more extensive hyperparameter tuning for the extremely low-resource case of English → Siswati. We tune the number of layers and the vocabulary size (see Figure 4.3), finding that smaller models (less layers) greatly improve validation performance. Our SSMT subwords have a

[5] https://huggingface.co/datasets/allenai/wmt22_african

[6] https://github.com/facebookresearch/fairseq/blob/main/fairseq/models/transformer/transformer_legacy.py

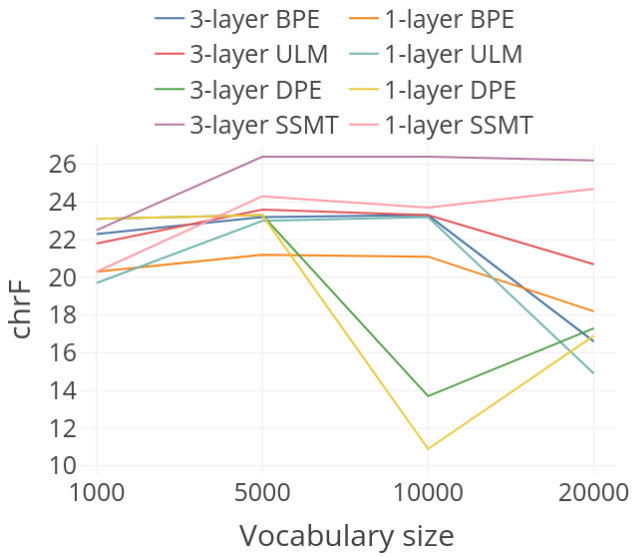

Figure 4.3: English → Siswati validation performance.

maximum segment length of 5 characters, since this is computationally feasible and validation performance does not improve with longer subwords.

4.5.3 Training and Decoding

We train all our models for 25 epochs initially and then continue training until validation performance stops improving for 5 epochs. We generally find that SSMT requires more epochs of training than our baselines for validation performance to plateau. While tokenisation-based baselines often reach peak validation performance around 25 epochs, SSMT keeps improving in subsequent epochs. We train our DPE segmentation models for 20 epochs (following He et al. [56]), so DPE requires 20 epochs of training for the segmentation model, followed by 25+ epochs for the translation model. We tried sampling ULM segmentations during training for regularisation, but initial experiments showed that maximising segmentations led to better validation performance.

One significant disadvantage of SSMT, during training and testing, is its additional computational overhead. Each probability in the SSMT dynamic program (Equation 4.5) requires a softmax computation, so SSMT takes an order of magnitude (10×) longer to train than pre-segmented models. For example, on English → isiZulu, our BPE-based baseline

	Mixture beam search		Dynamic decoding	
Beam size	BLEU	chrF	BLEU	chrF
1	11.8	49.7	13.6	52.2
3	11.5	49.2	14.1	53.6
5	11.2	49.4	**14.5**	**53.8**
7	11.4	49.6	14.3	**53.8**
10	11.5	49.8	14.4	**53.8**

Table 4.3: English → isiZulu validation set performance of SSMT with dynamic decoding compared to standard beam search over the lexicon and character distributions of the SSMT mixture model. Applying standard beam search to SSMT results in poor performance, which justifies the introduction and added computational complexity of dynamic decoding.

trains for 24 hours, while SSMT trains for 10 days (both on a single NVIDIA A100 GPU). SSMT training times are similar to those of the DPE segmentation model.

Similarly, dynamic decoding is less efficient than tokenisation-based beam search. On our evaluation sets, dynamic decoding takes on average 15 seconds to translate a single sentence (as opposed to our baselines, which take 0.05 seconds per sentence). We did experiment with a more efficient beam search over the combined lexicon and character vocabularies of SSMT, but this results in much worse validation performance than dynamic decoding, as shown in Table 4.3. We use a beam size of 5 for dynamic decoding and for tokenisation-based beam search with our baselines, since this optimised validation performance.

4.6 Experiment 1: Bilingual Machine Translation

We use the FLORES [148, 49] development test set (1012 multi-parallel sentences) to evaluate our final models. We compare models with two automatic metrics – BLEU and chrF [115]. The chrF score is a character-based metric that is more suitable for morphologically rich languages than token-based metrics like BLEU [9]. As a result, we tune our models based on validation chrF and focus on chrF when discussing our test set results. MT performance is shown in Table 4.4. We perform statistical significance testing through paired bootstrap resampling [76]. In terms of chrF, SSMT outperforms or matches the top-performing baseline

Model →	BPE		ULM		DPE		SSMT	
English to ↓	BLEU	chrF	BLEU	chrF	BLEU	chrF	BLEU	chrF
isiXhosa	14.3	**53.2**	**15.0**	**53.3**	**14.9**	**53.3**	**15.0**	**53.5**
isiZulu	13.5	53.2	**13.7**	53.0	**14.2**	**53.7**	**14.2**	**53.7**
Finnish	**15.0**	**50.1**	**15.0**	49.6	**15.4**	**50.0**	**14.4**	**50.1**
Siswati	0.2	23.4	0.4	23.7	0.3	23.5	**0.7**	**26.2**
Setswana	**10.2**	**36.9**	**10.1**	35.5	9.1	34.6	**9.7**	36.5
Afrikaans	33.4	64.2	33.5	64.3	**34.6**	**65.0**	32.0	63.6

Table 4.4: MT test set performance (FLORES devtest). Underline indicates best BLEU and chrF scores, while **bold** indicates scores with differences from the best that are not statistically significant (p-value of 0.05)

for all four agglutinating conjunctive languages. The same holds for BLEU on all three Nguni languages in our experiments.

These results show that SSMT is an effective alternative to tokenisation-based NMT for translating into morphologically complex languages. They also corroborate our findings for language modelling from Chapter 3, which showed that subword segmental modelling leads to greater consistency across the Nguni languages. While baselines do match SSMT performance for individual translation directions, they are otherwise inconsistent. No single tokenisation-based model matches SSMT chrF across multiple agglutinative conjunctive languages. Among our baselines, DPE exhibits the most competitive performance. However, this comes at the cost of DPE requiring multiple training steps: training a DPE segmenter model, applying that to a corpus, and training an NMT model on the segmented corpus. SSMT has the benefit of being a single model for translation, segmentation, and generation.

On languages with shorter word forms (Setswana and Afrikaans), SSMT is outperformed by baselines. There is a sharp contrast between the relative performance of SSMT on the morphologically complex and morphologically simple languages. We conclude that SSMT is not justified for languages with shorter word forms. For such languages, applying subword segmentation during preprocessing is sufficient. In Tables 4.6 and 4.7 we present a examples isiXhosa and Siswati translations, respectively, comparing SSMT output to baselines.

Model	chrF
2 models to segment + translate with beam search	
+BPE vocab –char (DPE)	23.3
+lexicon –char (SSMT –char)	23.7
+lexicon +char (SSMT)	23.1
1 model with dynamic decoding	
+lexicon –char (SSMT –char)	26.2
+lexicon +char (SSMT)	**26.4**

Table 4.5: English → Siswati validation set performance.

Low-resource translation analysis SSMT improves performance most drastically for English → Siswati, which is distinct among the translation directions in being extremely data scarce. Tables 4.7 shows examples of Siswati translations generated by SSMT and baseline models, demonstrating that tokenisation-based models often degenerate into repetitive text. As shown in Table 4.4, SSMT improves English → Siswati chrF by 2.5 over the best tokenisation-based NMT model. Even though absolute BLEU and chrF scores remain relatively low for Siswati, compared to higher resourced translation directions, this still represents a notable performance improvement. This is not simply because of particular hyperparameter choices, because the finding holds across different hyperparameter settings (see Figure 4.5). To investigate the factors behind SSMT's success, we perform an ablation analysis on the different components of SSMT (shown in Table 4.5).

DPE learns a subword vocabulary with BPE, but this does not improve performance over the frequency-based lexicon of SSMT. Our results also show that when SSMT is used as an upstream segmenter, it performs similarly to DPE. The largest gains do not come from learning subword segmentation during training, but from using the same model directly during inference. This is made possible by dynamic decoding, which proves critical to the performance of SSMT. Having a single model for segmentation, MT, and generation leads to the best performance overall.

English → isiXhosa example (a)

Source	The Hershey and Chase experiment was one of the leading suggestions that DNA was a genetic material.
Target	Uvavanyo lweHershey kunye noChase yenye yeengcebiso eziphambili zokuba iDNA yayiyinto yemfuza.
SSMT	Uvavanyo lukaHershey kunye noChase lwalungenye yeengcebiso eziphambili zokuba i-DNA yayiyinto yemfuza.
BPE	Uvavanyo lukaHershey noChase lwalunye lweengcebiso eziphambili zokuba i-DNA yayiyinto yemfuza.
ULM	Uvavanyo lweHershey noChase lwaluyenye yezona ngcebiso ziphambili zokuba i-DNA yayiyinto yemfuza.
DPE	Uvavanyo lukaHershey kunye neChase lwalunye lweengcebiso eziphambili zokuba i-DNA yayiyinto yofuzo.

English → isiXhosa example (b)

Source	They all ran back from where the accident had happened.
Target	Bonke babaleka babuyela apho kwenzeke khona ingozi.
SSMT	Bonke babaleka apho kwenzeke ingozi.
BPE	Bonke babuyela apho le ngozi yenzeke.
ULM	Bonke babuya apho ingozi yenzeke khona.
DPE	Bonke babalekela apho le ngozi yenzeke khona.

English → isiXhosa example (c)

Source	After officials verify the voter's identity, the voter drops the envelope into the ballot box and signs the voting roll.
Target	Emva koba amagosa eqinisekise iincukaca zomvoti, umvoti ufake imvlophu kwibhokisi yovota futhi atyikitye iphepha lokuvota.
SSMT	Emva kokuba amagosa aqinisekise ubungqina bokuvota, umvoti uyawa kwibhokisi yokuvota aze abonise umqulu wokuvota.
BPE	Emva kokuba amagosa aqinisekise isazisi somvoti, umvoti uyanciphisa imvulophu ebhokisini lokuvota aze asayine umqulu wokuvota.
ULM	Emva kokuba amagosa aqinisekise isazisi somvoti, umvoti uwela umkhankaso kwibhokisi yokuvota kwaye asayine umqulu wokuvota.
DPE	Emva kokuba amagosa aqinisekise isazisi somvoti, umvoti uwela umvolontiya kwibhokisi yokuvota aze atyikitye umqulu wokuvota.

Table 4.6: English → isiXhosa model translations compared to target sentences, with correctly translated word forms highlighted.

English → Siswati example (a)	
Source	He added that “they should not, however, be asked to take on obligations that go beyond their development stage, responsibility and capabilities.”
Target	Wengete watsi “akukafaneli bacelwe kutsi batsatse sibopho lesingetulu kwemandla abo, imitfwalo yabo nemakhono."
SSMT	Wababuta watsi: “Akukafaneli kutsatsa imisebenti yabo, kutsi batsatse imisebenti yabo nemisebenti."
BPE	Lokusho kutsi, ‘Kodvwa akufanele akwati kutsi ‘sekutfutfukiswa kwendlela yekutfutfukisa nekutfutfukisa kwendlela yekutfutfukisa nekutfutfukisa nekutfutfukiswa kwendlela yekutfutfuka.’
ULM	Kodvwa wachubeka watsi: “Lomuntfu akazange atitsatsele kutsi atitsatsele tidzingo tabo.”
DPE	Akukafaneli kutsi “kufanele kube nguloku lokukwengetiwe, kepha akukafaneli kutsatfwa kwetimphawu tabo.”
English → Siswati example (b)	
Source	After officials verify the voter’s identity, the voter drops the envelope into the ballot box and signs the voting roll.
Target	Ngemuva kwekutsi tisebenti telukhetfo seticinisekisile kutsi labavotako bangibo yini, labavotile bafaka timvilophi emabhokisini ekuvota bese basayina ephepheni lekuvota.
SSMT	Ngemuva kwekusebentisa sivakashi selibhubhiso, lelibhokisi kanye netinhlanyelo letisetjentiselwa eNingizimu.
BPE	Emuva kwekuvota kwalabavoti labaphakanyiswa ngulolubandlululo lolubandlululo kanye nalolubandlululo lolubandlululo lolubandlululo lolubandlululo lolubandlululo.
ULM	Umgomo wemgomo welivoti, livoti lelisekelwe livoti lelisekelwe livoti lelisekelwe livoti lemalunga.
DPE	Ngemuva kwaloko, kuveta titfombe letiphatsimandla teluhlolo lwekuveta titfombe letiphatselene nalolubandlululo.
English → Siswati example (c)	
Source	“We will endeavour to cut carbon dioxide emissions per unit of GDP by a notable margin by 2020 from the 2005 level,”Hu said.
Target	“Sitozama kunciphisa kukhicitwa kwe-carbon dioxide ngeyunithi ye-GDP ngelizinga lelinakekako nga-2020 kusukela kulelizinga la-2005” Hu washo.
SSMT	“Sitawucedza kunciphisa kutsi sisebentise imisebenti lenge-2020,” kusho Simakadze.
BPE	“Sitawuchubeka siphelele kutsi sinyatselo lesingaka-2020202020202020 ngemasotja ekupheleni kwa-2020202020 kuya eHanoni ngemnyaka wa 2018.
ULM	“Sitawuphelelwa ikhadi le-Gooooxioxideox ye-2005”kusho kutsi i-2018.
DPE	“Asizange sifeze sibalekele emacembe lamakhulu lasitfupha ekupheleni kwaseGrgen Richtersveld,” kusho Simakadze.

Table 4.7: English → Siswati model translations compared to target sentences, with correctly translated word forms highlighted.

4.7 Experiment 2: Unsupervised Morphological Segmentation

Morphemes are the primary linguistic units of the Nguni languages. There is a large body of work showing that aligning subwords more closely to morphemes can improve MT performance [60, 139, 106, 132, 131]. As for SSLM in Chapter 3, we analyse to what extent the subwords learned by SSMT resemble morphemes by evaluating our trained SSMT models on unsupervised morphological segmentation. The task is fully unsupervised, since our SSMT models were developed to optimise validation MT performance and never have access to morphological annotations (they are trained on raw text). Our goal is to evaluate to what extent SSMT "discovers" morphemes as linguistic units.

For this experiment, we use data from the SADiLaR-II project [47], applying the preprocessing scripts of Moeng et al. [98] to extract surface segmentations. The dataset contains 146 parallel sentences in English and each of the three Nguni languages from our MT experiments, providing morphological segmentations for all words. To use SSMT as a segmenter we run the Viterbi algorithm to compute the highest scoring subword segmentation of a Nguni-language target sentence given the English source sentence. This setup differs from the unsupervised segmentation experiments presented in Chapter 3, where SSLM learned subword segmentations from monolingual Nguni-language data. In contrast, SSMT leverages source-target sentence pairs, with the Nguni-language subword segmentations depending on the English source sentence. This allows SSMT to learn segmentations that are conditioned on cross-lingual information, potentially improving alignment with morphemes, since learning a lexical or morpheme-level translation mapping might improve translation performance. We compare SSMT to the baseline tokenisers from our MT experiments (BPE, ULM, and DPE) to assess whether SSMT learns subword units that are more aligned with morphemes than those produced by subword tokenisation.

Table 4.8 reports precision, recall, and F1 for morpheme boundary identification. SSMT has greater F1 scores than any of the baseline tokenisers across all three languages, indicating that generally SSMT learns subword boundaries that are closer to morphological boundaries.

	isiXhosa			isiZulu			Siswati		
Model	P	R	F1	P	R	F1	P	R	F1
BPE	37.16	25.42	30.19	51.57	29.62	37.62	19.57	16.17	17.71
ULM	**61.22**	34.65	44.25	**63.70**	31.72	42.35	**52.48**	45.26	48.61
DPE	51.52	44.24	47.60	59.66	41.64	49.05	16.96	17.00	16.98
SSMT	49.55	**72.60**	**58.90**	52.87	**66.41**	**58.87**	47.47	**61.89**	**53.73**

Table 4.8: Morpheme boundary identification performance across all words in the morphologically annotated dataset.

	isiXhosa			isiZulu			Siswati		
Model	P	R	F1	P	R	F1	P	R	F1
BPE	18.04	14.23	15.91	24.51	17.52	20.43	9.13	3.77	5.33
ULM	**31.59**	22.51	26.29	31.47	20.88	25.10	**32.31**	13.72	19.26
DPE	28.82	26.16	27.43	33.01	26.36	29.31	7.97	3.72	5.08
SSMT	31.58	**41.50**	**35.87**	**33.81**	**39.57**	**36.46**	27.57	**15.49**	**19.83**

Table 4.9: Morpheme identification performance across all words in the morphologically annotated dataset. Morpheme identification measures how much overlap their is between the subwords in a particular segmentation and the morphemes of a word.

Table 4.9 shows similar results when evaluating the same task according to morpheme identification, indicating more overlap between SSMT subwords and morphemes.

SSMT achieves the highest recall for all three languages, but exhibits relatively lower precision. This shows that SSMT sometimes over-segments words, which we also found to be the case for SSLM in Chapter 3, so it seems to be a general feature of subword segmental modelling. However, SSMT achieves higher precision and lower recall than SSLM (see Table 3.7 in the previous chapter). The evaluation sets are not the same (for MT we required a parallel dataset), so we cannot compare SSLM and SSMT directly, but the consistent performance difference does seem to indicate that SSMT over-segments to a lesser extent than SSLM. Table 4.10 presents examples of isiXhosa target sentence segmentations. These examples clearly show that SSMT over-segments less than SSLM. The SSLM segmentation examples presented in the previous chapter (Tables 3.10 & 3.11) contain several instances of character-level segmentation. Optimising subword segmentation for MT leads to subword units more aligned with morphemes than subwords learned for language modelling.

(a) English → isiXhosa example (a)	
Source	The number of jobs created through a municipality's local economic development initiatives including capital projects.
Target Morphemes	I-nani le-mi-sebenzi e-vel-is-iw-e-yo ku-ma-nyathelo ama-tsha o-phuhliso lw-e-zo-qoqosho ku-masipala we-n-gingqi u-ku-quk-a nee-projekthi ze-kapitali.
SSMT	I-nani le-miseb-enzi eve-lisi-weyo kuma-nyath-elo ama-tsha ophu-hli-so lw-ezo-qo-qo-sho kuma-sip-ala we-ngi-ngqi ukuq-uka nee-pro-j-ek-thi zek-ap-ita-li-.
BPE	Inani lem-isebenzi e-vel-isiweyo kum-anyathelo am-atsha o-phuhl-iso lwe-zo-qoqosho kum-asip-ala w-eng-ingqi uku-quka ne-ep-rojekthi z-ek-ap-it-ali.
ULM	I-nani le-misebenzi e-ve-li-siweyo ku-manyathelo ama-tsha o-phuhliso lwe-zoqoqosho ku-masipala we-ngingqi uku-quka nee-projekthi ze-ka-pit-a-li-.
DPE	Inani l-em-isebenzi ev-el-isi-weyo kum-anyathelo am-a-tsha o-phuhl-iso lwe-z-o-qoqosho kum-asip-ala w-e-ng-ingqi u-ku-quka n-e-ep-rojekthi z-ek-ap-ital-i-.
English → isiXhosa example (b)	
Source	A two- hour group meeting with CBP facilitators, Ward Committee members and working groups established for each outcome.
Target Morphemes	I-n-tlanganiso ye-qela yee-yure ezim-bini na-ba-bhexeshi be-CBP, a-ma-lungu e-Komiti ye-Wadi na-ma-qela a-sebenz-a-yo a-misel-w-e nge-si-phumo nga-si-nye.
SSMT	Int-lang-aniso ye-qela yee-yure ezimb-ini nab-abh-ex-eshi be-CBP-, amal-ungu eK-om-iti ye-W-adi nama-qela asebe-nzayo amis-elwe ngesi-phumo ngas-inye-.
BPE	Int-langaniso ye-qela ye-eyure ezimbini naba-bhe-x-es-hi be-C-B-P-, amalungu eK-om-iti ye-W-adi nama-qela a-sebenzayo am-iselwe nges-iphumo ngas-inye-.
ULM	I-ntlanganiso ye-qela ye-eyure ezimbini naba-bh-ex-e-shi be-C-B-P-, amalungu -e-K-o-mit-i ye-W-a-di na-maqela -a-sebenzayo -a-miselwe nge-siphumo ngasinye-.-
DPE	I-nt-langaniso ye-qela ye-eyure ezi-m-bini na-b-a-bh-ex-esh i-b-e-C-B-P-, amalungu e-K-om-iti ye-W-adi n-ama-qela asebenza-yo am-iselwe ng-esi-phumo nga-s-inye-.
English → isiXhosa example (c)	
Source	Consider the following questions.
Target Morphemes	Qwalasel-a le mi-buzo i-landel-a-yo.
SSMT	Q-wala-sela le mi-buzo iland-elayo-.
BPE	Q-wala-sela le mi-buzo i-landela-yo.
ULM	Q-wa-la-sela le mibuzo i-landelayo-.
DPE	Q-wala-sel-a l emi buzo i-landelayo-.

Table 4.10: Subword segmentations compared to the annotated morphological segmentations of isiXhosa target sentences. Correctly identified morphemes are highlighted.

4.8 Experiment 3: Morphological Compositional Generalisation

SSMT segments words into subword units that resemble morphemes to a greater degree than tokenisation-based subwords. Its subword segmentation scheme is more consistent with linguistic principles. Now we turn to the inverse question – does SSMT compose words from subwords according to linguistic principles? Asked another way, does SSMT learn *morphological composition*, the ability to compose the meanings of words from their constituent morphemes? To investigate this we design an experiment aimed at testing morphological compositional generalisation.

Compositional generalisation is the ability to compose novel combinations from known parts [111, 43]. Recent works have investigated whether neural models are able to achieve such generalisation [83, 61, 72]. For example, Keysers et al. [71] test whether models can handle novel syntactic combinations of known semantic phrases. They construct train/test splits with similar phrase distributions, but divergent syntactic compound distributions. We adapt their approach to construct a test set with a similar morpheme distribution to the train set, but a divergent word distribution. This targets whether models can handle novel combinations of known morphemes (previously unseen words consisting of previously seen morphemes). Table 4.11 categorises our experiment according to the NLP generalisation taxonomy of Hupkes et al. [62].

Given a train and test set, our procedure extracts examples from the test set to form a test *subset* that differs systematically, in terms of morphological distribution, from the train set. Simply put, we control the degree to which models are tested on familiar words, as opposed to unfamiliar words not encountered during training. Evaluating models on novel word forms – where the constituent morphemes are known, but their combination is new – requires the model to generalise by composing the meaning of words according to morpho-grammatical rules. By controlling the magnitude of the difference between the train set and test subset, we can evaluate our models on test sets that require different levels of morphological compositional generalisation. Unlike the experimental setup of Keysers et al.

Motivation					
Practical	*Cognitive*		*Intrinsic*		*Fairness*
☑			☑		
Generalisation type					
Compositional	*Structural*	*Cross-task*	*Cross-language*	*Cross-domain*	*Robustness*
☑	☑				
Shift type					
Covariate	*Label*		*Full*		*Assumed*
☑					
Shift source					
Naturally occuring	*Partitioned natural*		*Generated shift*		*Fully generated*
	☑				
Shift locus					
Train–test	*Finetune train–test*		*Pretrain–train*		*Pretrain–test*
☑					

Table 4.11: GenBench evaluation card (https://genbench.org/) categorising our morphological compositional generalisation experiment according to the generalisation taxonomy of Hupkes et al. [62].

[71], our approach does not require a newly trained model for each train/test split. Instead, under the assumption that a model's train set is available, we can extract test subsets for evaluating the morphological composition of that model.

4.8.1 Compound Divergence

Keysers et al. [71] propose compound divergence as a metric to quantify how challenging it is to generalise compositionally from one dataset to another. We use it to sample a subset of a test set that diverges from the morphological compositional distribution of a training set, thereby requiring morphological compositional generalisation to maintain good performance.

To compute morpheme distributions we segment our train and test sets into morphemes with the trained morphological segmenters of Moeng et al. [98]. Following Keysers et al. [71], we refer to morphemes as *atoms*, and words as *compounds*. For a dataset T, we compute the distribution of its compounds $F_C(T)$ as the relative word frequencies, and the distribution of its atoms $F_A(T)$ as the relative morpheme frequencies. For a train set V and test set W we compute compound divergence $\mathscr{D}_C(V||W)$ and atom divergence $\mathscr{D}_A(V||W)$, respectively

quantifying how different the word and morpheme distributions of the train and test sets are (larger divergence implies greater difference). We implement a procedure for extracting a subset of the test set such that $\mathscr{D}_C$ can be specified and $\mathscr{D}_A$ is held as low as possible, producing a test set that requires models trained on V to generalise to new morphological compositions.

We use the definitions of compound and atom divergence proposed by Keysers et al. [71]. For a train set V and test set W, we compute the compound divergence and atom divergence, respectively as

$$\mathscr{D}_C(V||W) = 1 - C_{0.1}(F_C(V)||F_C(W)), \tag{4.10}$$

$$\mathscr{D}_A(V||W) = 1 - C_{0.5}(F_A(V)||F_A(W)), \tag{4.11}$$

where $C_\alpha(P||Q)$ is the Chernoff coefficient [21]. This is a measure of the similarity of two distributions P and Q, computed as

$$C_\alpha(P||Q) = \sum_k p_k^\alpha q_k^{1-\alpha}, \tag{4.12}$$

where α is a parameter that weighs the importance of the distributions in the similarity metric. We follow Keysers et al. [71] in setting $\alpha = 0.1$ for compound divergence (it is more important to measure whether or not compounds occur in the train set than to measure how close the distributions are) and $\alpha = 0.5$ for atom divergence (atom distributions should match as far as possible).

We implement a procedure that, given a train set V, extracts a prespecified number of sentences from a test set W, such that $\mathscr{D}_C(V||W) = \mathscr{D}_C^{\text{target}}$ (where $\mathscr{D}_C^{\text{target}}$ is the desired compound divergence) and $\mathscr{D}_A(V||W)$ is held as low as possible. The procedure starts with the empty test subset and iteratively adds one sentence from the test set. At each step, it randomly samples k sentences from the test set (we set $k = 100$) and adds the sentence that minimises

$$|\mathscr{D}_C - \mathscr{D}_C^{\text{target}}| + \mathscr{D}_A, \tag{4.13}$$

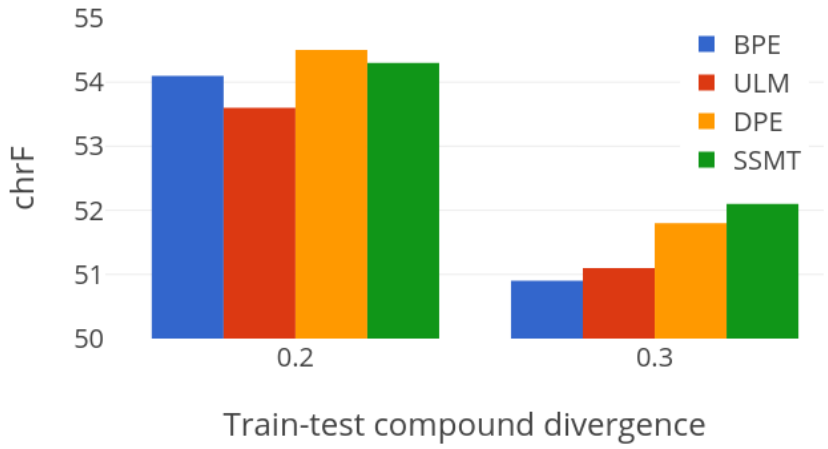

Figure 4.4: MT performance of our English → isiZulu models on test subsets that are easier (left) and harder (right) in terms of morphological compositional generalisation.

where $\mathscr{D}_C^{\text{target}}$ is the prespecified compound divergence target for the experiment. Iteratively adding sentences that minimise Equation 4.13 results in a test subset containing atoms (morphemes) that the model was exposed to during training, but compounds (words) that it has not encountered, or at least not very frequently. We can control the degree of compositional novelty in the test subset compounds by setting $\mathscr{D}_C^{\text{target}}$ in our procedure.

4.8.2 Results

For this experiment we focus on English → isiZulu translation. We extract two test subsets of 300 sentences each from isiZulu FLORES devtest. For the first subset we specify $\mathscr{D}_C^{\text{target}} = 0.2$, while for the second $\mathscr{D}_C^{\text{target}} = 0.3$. We settled on these values since it was not possible, using our iterative sampling procedure, to extract test subsets that increased compound divergences beyond this range, while maintaining a low atom divergence ($\mathscr{D}_A$ was held to around 0.07 for both splits). The result is two test subsets that require varying degrees of morphological generalisation. The subset with $\mathscr{D}_C = 0.3$ is more challenging than the $\mathscr{D}_C = 0.2$ subset, provided the model is trained on the same train set as ours (English-isiZulu

WMT22 dataset). We reuse the English → isiZulu models trained for our MT experiments, evaluating them on our newly extracted test subsets.

The results are shown in Figure 4.4. On the less challenging subset ($\mathscr{D}_C = 0.2$), DPE slightly ouperforms SSMT, while the average chrF score of the four models is 54.1. On the more challenging subset ($\mathscr{D}_C = 0.3$), the average chrF score drops to 51.5, which shows that models cannot maintain the same level of performance in the face of novel morphological compositions. This points to the fact that NMT models are not reliably learning morphological composition but are instead partially relying on linguistically invalid heuristics and memorising word forms encountered during training. SSMT proves to be most robust to the distributional shift, achieving the best chrF score on the more challenging subset. This suggests that SSMT is learning composition more aligned with morphological composition, enabling it to handle novel morphological combinations more effectively. This is likely connected to our observation that SSMT learns subwords closer to morphemes, as modelling the Nguni languages with linguistically aligned subword units would facilitate linguistically sound subword-to-word composition. SSMT and DPE comfortably outperform BPE and ULM, indicating more generally that learning subword segmentation during training improves morphological compositional generalisation.

4.9 Limitations

The main downside of SSMT (compared to tokenisers like BPE and ULM) is its computational complexity. Our architecture (Figure 4.1) introduces additional computation in two ways. Firstly, the decoder conditions on the character-level history of the target sentence, so it processes more tokens than subword-based decoders. Secondly, the dynamic programming algorithm (Equation 4.5) requires more computations than tokenisation-based NMT training. In practice, SSMT takes an order of magnitude ($10\times$) longer to train than BPE and ULM. Dynamic decoding also adds computational complexity to testing, although this is less of an issue since test set sizes usually permit run times within a few hours.

It would depend on the practitioner to decide whether the performance boosts obtained by SSMT justify the longer training and decoding times. However, the problem is somewhat alleviated by the fact that SSMT is particularly strong for data scarce translation. In such settings, where training sets are comparatively small, training times remain relatively short despite the higher computational complexity. For instance, training SSMT for English → Siswati translation takes less than a day on subpartitions of an A100 GPU.

4.10 Conclusion

In this chapter we proposed SSMT, a neural sequence-to-sequence model that unifies MT and subword segmentation. We also proposed dynamic decoding, a decoding algorithm that enables text generation with trained subword segmental models. We train and evaluate SSMT and tokenisation-based NMT models for six translation pairs: English → (isiXhosa, isiZulu, Siswati, Finnish, Setswana, Afrikaans). These languages span three morphological typologies and several levels of data availability, so they provide a varied test suite to evaluate subword methods across different linguistic contexts. SSMT outperforms or matches the top-performing baselines on languages that are agglutinating and conjunctively written, including all three Nguni languages in our experiments (isiXhosa, isiZulu, and SiswatI). Is is outperformed by tokenisation-based NMT on languages that are morphologically simpler and disjunctively written. SSMT achieves its biggest gains on English → Siswati, which is our most data scarce translation direction. We conclude that SSMT is justified for morphologically complex languages and especially useful for languages that are both morphologically complex and under-resourced.

As for SSLM, we analyse to what extent SSMT is learning (without explicit supervision) to model the complex morphological structure of Nguni languages. We again analyse the linguistic plausibility of SSMT subwords through the task of unsupervised morphological segmentation, showing that SSMT learns subwords that are closer to morphemes than those of NMT subword tokenisers. To test whether SSMT is able to reliably compose the meaning of words from their constituent morphemes, we adapt the methods of Keysers et al. [71]

to construct an English → isiZulu test set for morphological compositional generalisation – the ability to generalise to previously unseen combinations of known morphemes. The performance of all our models degrade on the more challenging test set, but SSMT exhibits the greatest robustness. We posit that SSMT's performance gains on morphologically complex languages are partially due to its morphologically consistent segmentations and its superior modelling of morphological composition. In summary, this chapter presents the following contributions:

1. We propose SSMT, which unifies subword segmentation and MT in a single end-to-end neural architecture.

2. We propose dynamic decoding, a segmentation-aware text generation algorithm for subword segmental models.

3. We train bilingual SSMT models for six translation directions and find that it outperforms tokenisation-based NMT for translating from English to Nguni languages.

4. We evaluate SSMT as an unsupervised morphological segmenter, where it outperforms MT subwords tokenised by BPE, ULM, and DPE.

5. We evaluate SSMT on a train/test split that requires morphological compositional generalisation, where it proves more robust to the distribution shift than existing subword-based models.

SSMT matches or improves performance over tokenisation-based NMT for translating from English to Nguni languages, and proves especially effective for the extremely low-resource case of English to Siswati translation. We replicate our findings from the previous chapter that subword segmental modelling learns subwords that are closer to morphemes. We also show that SSMT generalises more robustly to a held-out evaluation set designed to test morphological compositional generalisation. These results build on our findings from the previous chapter to provide further evidence that subword segmental models acquire more knowledge about morphological systems than standard tokenisation-based models.

Importantly, with the introduction of dynamic decoding, this chapter has established the benefits of subword segmental text generation for the low-resource Nguni languages. It is natural to wonder whether scaling subword segmental training would lead to further performance gains, but this question might not be practically relevant given the under-resourced nature of the Nguni languages. However, there is one way in which we can further scale SSMT given the available datasets – by extending it to multilingual modelling, which we do in the following chapter.

5

Multilingual Subword Segmental Machine Translation

In the previous chapter we showed that subword segmental machine translation (SSMT) improves bilingual translation for Nguni languages. In this chapter we investigate combining SSMT with multilingual modelling. We train a single SSMT model to translate between multiple languages, thereby learning target-side subword segmentation to optimise performance across multiple translation directions. We train multilingual SSMT models, selectively varying the languages modelled together to account for linguistic relatedness. Alongside each SSMT model, we train a range of multilingual tokenisation-based models. This experimental setup allows us to systematically compare the efficacy of several prominent subword methods in inducing cross-lingual transfer. Consequently, this chapter goes beyond simply applying SSMT in the multilingual setting and serves as a more general, in-depth investigation into the role of subwords in multilingual MT. We find that SSMT is not advantageous for multilingual modelling, but improves performance of models finetuned for Nguni translation after pretraining on an unrelated language. More generally, our study confirms that decisions around subword methods are key to optimising the benefits of multilingual MT. Subword regularisation boosts synergy in multilingual models, whereas BPE more effectively facilitates cross-lingual finetuning. Notably, our results also suggest that differences in orthographic word boundary conventions pose a significant barrier to cross-lingual transfer.

5.1 Motivation

In the previous chapter we established that SSMT can improve bilingual translation for low-resource, agglutinative languages with conjunctive orthographies. It achieves especially large performance gains in the extremely low-resource setting of English → Siswati translation. Given such strong performance for low-resource languages, it is natural to ask whether SSMT can be used in conjunction with other approaches to low-resource MT, such as multilingual modelling. Multilingual modelling boosts MT for low-resource languages through cross-lingual transfer from high-resource languages. In training one encoder-decoder to translate between multiple languages, the model can leverage similarities between languages and apply some of its knowledge acquired from high-resource languages to low-resource languages. This is partly achieved through shared subword representations. By exploring SSMT in the multilingual setting, we aim to determine if *learning* multilingual subword segmentation during training can optimise cross-lingual transfer.

5.1.1 Subword Segmentation and Cross-Lingual Transfer

The conventional approach to subword segmentation for multilingual modelling is to apply tokenisers like BPE [135] and ULM [81] to the joint multilingual training corpus. This produces a single multilingual subword vocabulary used by the model for all languages. This shared subword vocabulary is one of the mechanisms that enable cross-lingual transfer. During training, the model learns subword representations that encode syntactic and semantic knowledge, some of which is transferable across languages (e.g. cognates, loan words, named entities). Given that these subword representations are shared among languages, the model can learn features from high-resource languages that enhance its features for low-resource languages and improve translation quality.

This interaction between subword representations and cross-lingual transfer is why decisions around subword segmentation and subword vocabulary could potentially affect the performance of multilingual models. This phenomenon has been studied in previous works. Subword segmentation has been shown to affect cross-lingual transfer through factors such

as the amount of cross-lingual subword overlap [113, 167, 112] and the under-representation of low-resource languages in the vocabulary [161, 176]. These factors have mainly been studied for multilingual language modelling [128, 89, 20], but the same concerns hold for MT [158].

In this chapter we investigate how the subword modelling approach of SSMT, which learns subword segmentations that optimise MT performance, fares in the multilingual setting. To test this we train multilingual SSMT models and compare their performance to multilingual models based on standard tokenisers, as well as tokenisation methods designed specifically to enhance cross-lingual transfer. We run several such comparisons, in each instance varying the languages modelled together to analyse the relationship between language similarities and cross-lingual transfer. While our main motivation for running these experiments is to test the multilingual capabilities of SSMT, our experimental setup allows us to examine the relationship between subword segmentation, linguistic typology, and cross-lingual transfer more generally. Taking a broader view of our investigation, it serves as a systematic analysis of subwords and cross-lingual transfer in multilingual translation. Our study is the first to compare a representative selection of the many existing subword methods in the context of multilingual MT. We do so through the lens of *synergy* and *interference*, which allow us to quantify how well different subword methods handle the tradeoffs of multilingual modelling.

5.1.2 Synergy and Interference

Machine translation (MT) models have become increasingly multilingual [27]. This greatly benefits low-resource languages through positive transfer from high-resource languages [53, 4]. However, increasing multilinguality in a limited shared parameter space can lead to suboptimal performance for high-resource languages [42, 148]. There is a tradeoff between maximising positive cross-lingual transfer (also known as synergy) while minimising negative cross-lingual interaction (also known as interference). Several modelling decisions affect synergy and interference in multilingual MT, including multilingual subword segmentation. The shared subword vocabulary of multilingual models presents a similar trade-off as the shared parameter space – overlapping subword representations induce synergy, but having

Language	Family	Morphology	Orthography	*What is your name?*	*Thank you!*
Siswati (ss)	Bantu/Nguni	agglutinative	conjunctive	Ngubani ligama lakho?	Ngiyabonga!
isiXhosa (xh)	Bantu/Nguni	agglutinative	conjunctive	Ungubani igama lakho?	Enkosi!
Setswana (ts)	Bantu/Sotho-Setswana	agglutinative	disjunctive	Leina la gago ke mang?	Ke a leboga!
Afrikaans (af)	Germanic	analytic	disjunctive	Wat is jou naam?	Dankie!

Table 5.1: We vary the language modelled alongside Siswati to control relatedness, morphology, and orthography.

to represent multiple languages in a limited vocabulary can impede performance for certain translation directions [20, 128, 112].

Synergy and interference are well-established phenomena [42, 4, 148], but not well understood. Shaham et al. [136] address this by systematically analysing the role of several factors in synergy and interference: (1) model size, (2) data size, (3) language proportions, (4) number of languages, and (5) language relatedness. Their results show that scaling model size and tuning the data sampling temperature (which controls the proportion of data from different languages during training) greatly alleviates interference. One aspect their study fails to consider is subword segmentation. They do not vary subword segmentation in their experiments, using the same sentencepiece [82] vocabulary across all models. In this chapter, we adapt their experimental methodology to analyse the role of subword segmentation in the synergy/interference tradeoff.

5.2 Methodology

Siswati serves as the low-resource target language in our study, kept constant across all our experiments. We systematically vary two factors in our models – the subword method and language modelled alongside Siswati. This allows us to compare how different subword approaches interact with typologically varied languages in multilingual/cross-lingual settings.

The linguistic diversity of South Africa is an ideal testing ground for our purposes. Siswati is a low-resource, morphologically complex language, so effective subword modelling is critical for dealing with the inevitably high proportion of out-of-vocabulary words in held-out datasets. We alternate the higher resourced language modelled alongside Siswati between

isiXhosa, Setswana, and Afrikaans. Table 5.1 shows how these language present varying linguistic relationships to Siswati. IsiXhosa is closely related with some overlapping vocabulary. Setswana is somewhat related and also agglutinative, but diverges in its orthography - its writing system is disjunctive [117]. This refers to how linguistic words (e.g. nouns, verbs) are represented as orthographic words (space-separated tokens). Disjunctive orthographies write a single linguistic word as multiple orthographic words (e.g. in Setswana prefixal morphemes are space-separeted from verbal roots).

While linguistic relatedness and morphological complexity are obvious features to consider in any analysis of cross-lingual interactions, we are unaware of work considering the impact of orthographic word boundary conventions. We include it as a factor in our study because of its potential relevance to subword segmentation. Orthographic word boundaries determine the pre-tokenisation of text before subword segmenters are applied, so it could well affect aspects like segmentation granularity and overlap between the subword vocabularies of different languages. Afrikaans is linguistically unrelated to Siswati and also disjunctive, but because of its analytic morphology (lower morpheme-to-word ratio) its written words are sometimes more aligned to those of Siswati (e.g. Table 5.1 shows example phrases for which the Siswati word counts are closer to Afrikaans than Setswana).

This selection of languages allows us to isolate the cross-lingual effects of linguistic relatedness, morphological typology, and orthographic word boundary conventions. In the case of Setswana–Siswati, we can study whether the potentially positive cross-lingual effect of their linguistic relatedness is limited by the fact that the two languages have very different conventions for orthographic word boundaries.

5.2.1 Multilingual Modelling

We train two bilingual models and one trilingual model per language pair (see Table 5.2). This setup allows us to compare how MT performance changes for en→ss and en→xh/ts/af going from bilingual models to multilingual models. Following Shaham et al. [136], we measure synergy/interference for a translation direction $s \rightarrow t$ by the relative difference in

Languages	Examples	Subwords
en→ss	166k	BPE/ULM/SSMT
en→xh/ts/af	1.6m	BPE/ULM/SSMT
en→ss+xh/ts/af	1.6m+166k	BPE/ULM/SSMT/OBPE

Table 5.2: Multilingual experimental setup: bilingual and trilingual models (bilingual OBPE is equivalent to BPE).

performance between a bilingual model trained to translate only from s to t and a multilingual model trained to translate to an additional language.

Shaham et al. [136] use test set cross-entropy loss to measure MT performance, but this cannot be compared across different subword segmentations. Instead, we use test set chrF++ [116] to measure performance. It is a reference-based metric that combines word and character information, so it is well suited to evaluate MT performance across different morphological typologies. Our modified formula for measuring synergy/interference is

$$I_{s\to t} = \frac{\text{CHRF++}(M^{\text{multi}}_{s\to t}) - \text{CHRF++}(M^{\text{bi}}_{s\to t})}{\text{CHRF++}(M^{\text{bi}}_{s\to t})},$$

where M are trained multilingual/bilingual models evaluated on $s \to t$ translation. Negative values of $I_{s\to t}$ indicate worse performance for $s \to t$ in the multilingual model (interference) and positive values indicate improved performance (synergy).

5.2.2 Cross-Lingual Finetuning

We train a bilingual subword segmenter and MT model for en→xh/ts/af, and then finetune and evaluate the model in the other translation directions (e.g. pretrain en→xh and finetune on en→ss, en→ts, and en→af). Varying the subword method reveals how different subwords facilitate cross-lingual transfer between different language typologies, and from higher resourced languages (isiXhosa/Setswana/Afrikaans) to lower resourced Siswati.

5.2.3 Adapting SSMT for Multilingual Translation

Multilingual SSMT is a straightforward adaptation of SSMT for multilingual modelling. The architecture and training algorithm are identical to those proposed in the previous chapter and our dynamic decoding algorithm can still be used to generate translations. Multiple translation directions are incorporated, as commonly done in multilingual MT, by appending special tokens to the start of each source sentence during training and evaluation to indicate source and target languages (e.g. "en – xh" to indicate translation from English to isiXhosa). This produces an end-to-end NMT model trained to translate between multiple languages while jointly learning a single subword segmentation scheme (for all target languages) that optimises its multilingual MT training objective.

5.2.4 Baselines

We compare SSMT to four multilingual subword tokenisers (three per experiment). Our baselines represent the main paradigms of subword segmentation - deterministic segmentation, subword regularisation, and subword techniques for enhancing cross-lingual transfer.

1. **BPE** [135] iteratively adds frequently co-occurring subwords to its vocabulary, based on subword frequency in the joint, multilingual training corpus. We use it as a deterministic segmenter.

2. **ULM** [81] learns segmentation to optimise the likelihood of the joint, multilingual training corpus under a unigram language model. It can be used as a probabilistic segmenter, exposing models to varying subword segmentations (for regularisation) by sampling from the ULM distribution. Sampling is controlled by smoothing parameter, which we set to 0.5 following validation-based tuning.

3. **Overlap BPE (OBPE)** [112] modifies the BPE merge criterion to account for language vocabulary representation and boost subword overlap among languages in multilingual vocabularies. We adopt it as the third baseline in our multilingual experiments to see if increased shared subword representations promote synergy.

4. **EXTEND (XBPE)** [163] extends the BPE vocabulary of a pretrained MT model by adding BPE subwords from a new translation direction to the initial subword vocabulary. The new subword embeddings are randomly initialised and learned during finetuning. We adopt it as the third baseline in our finetuning experiments to see if the vocabulary extension enhances cross-lingual transfer.

5.3 Experimental Setup

We extend our SSMT implementation in the fairseq library[1] to multilingual modelling and publicly release all our code.[2] We now present the datasets and model configurations used in our experiments.

5.3.1 Data

We use the WMT22 dataset [2] for training, and the FLORES dataset [49, 148] for validation and testing. The sizes of our training sets (number of parallel sentences) are shown in Table 5.2. For en→ss this is the full WMT22 dataset, but for en→xh/ts/af we sample sentences from en→xh and en→ts to match the size of en→af, removing data size as an influence. We also removed example from en→xh/ts/af where English source sentences were found in en→ss to neutralise the positive transfer effect of multi-parallel overlap [142].

5.3.2 Hyperparameters

We use the model size and training hyperparameters of the fairseq transformer-base architecture[3] (6 encoder layers, 6 decoder layers). When applying subword methods we specify a shared vocabulary size of 8k. This is slightly smaller than 10k, which was found to be optimal for these datasets in Chapter 4, since we use smaller subsets of the WMT22 datasets.

[1] https://github.com/facebookresearch/fairseq

[2] https://github.com/francois-meyer/ssmt

[3] https://github.com/facebookresearch/fairseq/blob/main/fairseq/models/transformer/transformer_legacy.py

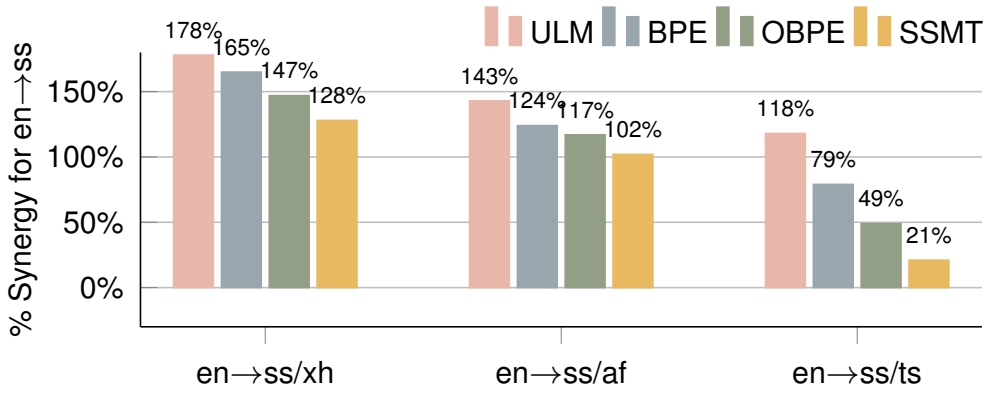

Figure 5.1: Performance increase for English→Siswati through multilingual modelling varies greatly across subword methods and linguistic contexts.

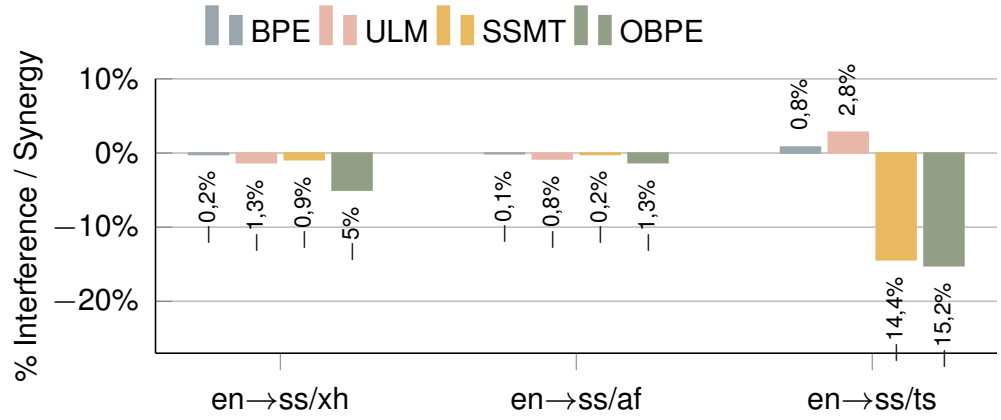

Figure 5.2: Performance change for English→isiXhosa/Afrikaans/Setswana through multilingual modelling alongside English→Siswati.

5.3.3 Training and Decoding

We train our models for 45 epochs and use a language sampling temperature of $T = 1.5$, where $T > 1$ increases the representation of low-resource languages during training. In our cross-lingual finetuning experiments we finetune models on en→ss for 20 epochs and on en→xh/af/ts for 10 epochs. To generate translations with our multilingual SSMT models, we use the dynamic decoding introduced in the previous chapter with a beam size of 5. For all our baselines, we similarly use a beam size of 5 in the beam search decoding.

Model	tgt	BPE	ULM	SSMT	OBPE
en→ss/xh	ss	33.4	**35.1**	33.6	31.1
	xh	46.8	**47.2**	46.1	44.6
en→ss/af	ss	28.2	**30.6**	29.7	27.4
	af	60.9	**61.1**	60.2	60.2
en→ss/ts	ss	22.5	**27.5**	17.8	18.8
	ts	31.3	**34.7**	23.6	26.4

Table 5.3: Test set chrF++ of trilingual models.

5.4 Results & Discussion

We plot the synergy/interference analysis of our multilingual experiments in Figures 5.1 and 5.2, while the absolute performance of the models are provided in Table 5.3. The results from our cross-lingual finetuning experiments are presented in Table 5.4.

5.4.1 Which subwords promote synergy and minimise interference?

ULM consistently achieves greater synergy than the other subword methods. This holds across all linguistic contexts (Figure 5.1) and results in better absolute performance for all translation directions (Table 5.3). It comes at the cost of minimal interference for the higher resourced languages (Figure 5.2). The deterministic segmenters, BPE and OBPE, do induce synergy, but fail to match the multilingual performance of ULM. The subword regularisation of ULM ensures that models are more robust to the varied subwords of multilingual modelling.

SSMT exhibits the lowest synergy among all models, with significant interference for the high-resource languages. The results suggest that SSMT is unable to learn a single subword segmentation scheme that is optimal across multiple translation directions. In trying to satisfy the diverging subword segmentation requirements of multiple languages, SSMT learns suboptimal subwords for both low-resource and high-resource languages. This is supported by the finding that SSMT performance degrades most dramatically in the case

BPE				
	xh	af	ts	ss
xh	–	**60.4**	**38.7**	**32.3**
af	46.4	–	**36.9**	28.3
ts	**43.8**	**58.5**	–	**25.5**

ULM				
	xh	af	ts	ss
xh	–	15.8	1.9	31.1
af	13.5	–	17.3	28.2
ts	10.8	3.6	–	24.7

SSMT				
	xh	af	ts	ss
xh	–	59.9	38.0	32.2
af	**46.6**	–	36.3	**30.1**
ts	42.0	56.8	–	23.4

XBPE				
	xh	af	ts	ss
xh	–	59.5	37.3	29.9
af	46.1	–	35.6	25.0
ts	43.4	58.3	–	21.9

Table 5.4: Test set chrF++ of pretraining for en→xh/af/ts (rows) and finetuning on en→xh/af/ts/ss (columns). Diagonal entries are bilingual models with no finetuning.

of English → Setswana + Siswati, which highlights its inability to effectively reconcile the diverging orthographic word boundary conventions of the two languages.

5.4.2 Which subwords transfer cross-lingually?

BPE exhibits the greatest cross-lingual transferability across the majority of translation directions in our experiments. In contrast to our multilingual findings, the subword regularisation of ULM proves a barrier to cross-lingual finetuning. ULM is a probabilistic segmenter that is sampled during training, but when the probabilistic model is based on one language and applied to another, its samples might yield highly inadequate subword units. The consistent deterministic segmentation of BPE allows the finetuned model to adapt to a new translation direction effectively, while the vocabulary extension of XBPE fails to aid finetuning.

SSMT is competitive for most finetuning scenarios in our experiments and there are two instances where SSMT achieves the top-performing results, both of which involve transfer to Nguni languages. SSMT outperforms all tokenisation-based models in the case of pretraining for English → Afrikaans, and subsequent adaptation to English → isiXhosa and English → Siswati. Afrikaans is linguistically unrelated to the Nguni languages, which makes

cross-lingual transfer particularly challenging. In tokenisation-based NMT, the subword tokenisation from Afrikaans pretraining remains fixed during Nguni-language finetuning, resulting in a mismatch: Afrikaans subword units are used to train Nguni MT models. SSMT is not constrained to the subword units from pretraining, as it can "finetune" its subword segmentation to optimise the new translation objective. Because it can reconfigure its subword segmentation during cross-lingual finetuning, SSMT adapts more effectively from the simple morphology of Afrikaans to the complex morphological systems of Nguni languages. Subword segmental finetuning offers a distinct advantage over tokenisation-based finetuning, particularly when adapting pretrained models to low-resource languages with complex morphological structures.

5.4.3 What is the role of linguistic typology?

A consistent pattern emerges in the cross-lingual dynamics between Siswati and other languages. IsiXhosa modelling proves to be most beneficial for Siswati performance. Afrikaans achieves less transfer, presumably because it belongs to a language family that is unrelated to the Nguni languages. Somewhat surprisingly, the weakest synergy is between Siswati and Setswana, even though both are agglutinative Bantu languages. This highlights the impact of orthographic systems on cross-lingual transfer: *diverging word boundary conventions can impede cross-lingual transfer more than linguistic unrelatedness*. Data-driven multilingual models that learn from text might miss underlying similarities between languages that are obscured by superficial differences in their surface realisations.

Our results highlight two interacting effects. Firstly, linguistic relatedness does play a role – Siswati consistently benefits more from isiXhosa than from Setswana or Afrikaans. Secondly, in the specific case of Setswana–Siswati, their relatedness is rendered all but irrelevant by the fact that the languages have diverging orthographies. Afrikaans does not have the extremely low morpheme-to-word ratio of Setswana so even though it is less related to Siswati than Setswana, the disjunctive orthography of Setswana prevents transfer to Siswati. Linguistic distance plays a role in both cases (Afrikaans–Siswati and Setswana–Siswati) but for Setswana-Siswati it is a less important factor than orthography.

Orthography is a notable difference between Nguni languages like Siswati and Sotho-Tswana languages like Setswana [117]. Taljard and Bosch [146] show that the diverging orthographies of these two language groups necessitate different approaches to traditional NLP tasks, even though the languages are linguistically and morphologically related. Our results suggest a similar situation for cross-lingual transfer in multilingual modelling: differences in orthographic word boundary conventions harms synergy between otherwise related languages.

5.5 Limitations

This chapter's study is limited to translation from English to four South African languages. While the chosen languages are typologically diverse, our conclusions might not necessarily hold for languages from other language families, given the wide range of orthographic systems across languages. The relative performance of subword methods across languages is relatively consistent in our results, so our interpretation of the results are well supported. However, a more detailed analysis on the interaction between choice of subword segmentation method and linguistic typology could yield additional explanations of the results.

5.6 Conclusion

In this chapter we extended SSMT to multilingual modelling, unifying subword segmentation and translation between multiple languages in a single end-to-end trainable model. We evaluate it alongside establish subword methods in small-scale multilingual experiments. Our experiments aim to determine which subword methods most benefit the Nguni languages, with a particular focus on the extremely low-resource case of English $\rightarrow$ Siswati, which stands to gain the most from cross-lingual transfer. Our study involves two sets of MT experiments that reflect common strategies for low-resource MT – multilingual modelling and cross-lingual finetuning to adapt models to new languages.

In our multilingual experiments we extend the methodology of Shaham et al. [136] to investigate how the choice of subword method affects synergy and interference in multilingual modelling. We train bilingual and multilingual models (trilingual models that translate from English to Siswati and another language) and quantify synergy/interference by measuring performance change from bilingual to multilingual modelling. We find that multilingual SSMT does induce synergy, consistently improving over bilingual SSMT for English $\rightarrow$ Siswati. However, multilingual SSMT does not surpass the top-performing tokenisation-based baselines. In fact, among all the methods we tested, subword regularisation (ULM) consistently facilitates the greatest synergy at the cost of of negligible interference.

In our cross-lingual finetuning experiments we train bilingual MT models and subsequently adapt them for new target languages. This allows us to compare the ability of different subword methods in facilitating cross-lingual transfer during finetuning. In contrast to our multilingual results, ULM is the worst-performing method for cross-lingual adaptation. Overall, BPE achieves the best results on average after cross-lingual finetuning. However, SSMT improves adaptation between linguistically unrelated languages, likely because of its ability to adapt its subword segmentation to a new morphological system during cross-lingual finetuning. SSMT outperforms all baselines in the case of adapting an English to Afrikaans model, via finetuning, to English to Nguni translation (isiXhosa and Siswati).

In both sets of experiments we systematically vary the languages modelled alongside each other to analyse how the linguistic relationship between interacting languages affects cross-lingual transfer. We train models to translate from English to a selection of four South African languages – Siswati, isiXhosa, Setswana, and Afrikaans. These languages were chosen to cover a variety of linguistic typologies (see Table 5.1) that we hypothesise to be relevant to the cross-lingual transfer induced by shared subword representations, namely language relatedness, morphology, and orthographic word boundary conventions (the degree to which morphemes are concatenated or written as separate orthographic words). Our experiments yield intriguing results regarding this research question. While related languages indeed benefit more from cross-lingual transfer, we also show that differences in orthographic

word boundaries can impede cross-lingual transfer between languages that are otherwise related (Siswati and Setswana).

In summary, this chapter presents the following contributions:

1. We train multilingual SSMT models that unify subword segmentation and multilingual MT in an end-to-end architecture.

2. We present a synergy/interference analysis of subword methods for multilingual MT, showing that subword regularisation (ULM) best facilitates cross-lingual transfer to Siswati with little interference for higher resourced languages.

3. In finetuning experiments we show that BPE best facilitates cross-lingual adaptation overall, while SSMT outperforms BPE on transfer from the morphologically simpler Afrikaans to Nguni-language translation.

4. We selectively vary the linguistic typology of languages modelled together, demonstrating that linguistic relatedness can enhance transfer but is sometimes impeded by differences in orthographic word boundaries.

Unlike in our previous chapter, SSMT does not improve over the best performing tokenisation-based models for multilingual MT. In cross-lingual finetuning experiments, SSMT does improve performance when models are adapted from Afrikaans to Nguni languages. We therefore conclude that SSMT is not justified for multilingual modelling, but does show promise for adapting models to divergent languages through subword segmental finetuning. Our experimental setup also produces results that are relevant beyond the main focus of this thesis, serving as a more general linguistic-oriented investigation into the role of subword segmentation in multilingual MT. Our findings in this regard confirm the significant influence of decisions around subword segmentation on cross-lingual transfer. ULM proves optimal for transfer to low-resource languages in multilingual modelling, while BPE enables greater cross-lingual transfer during finetuning. Cross-lingual transfer also depends on the linguistic typologies of interacting languages.

Besides language relatedness, we show that similarities/differences in orthographic word granularity can greatly affect multilingual performance. The role of orthographic word

boundary conventions has not been studied much in NLP and our results suggests that it warrants further investigation. The misaligned word boundaries of the Nguni and Sotho-Tswana languages, which are otherwise linguistically related, provides an opportunity for such work. Future research could aim to design multilingual techniques (e.g. subword methods or neural architectures) that see past orthographic differences in order to leverage more fundamental similarities between languages.

6

Subword Segmental Pointer Generator

So far this thesis has introduced subword segmental models for two tasks, namely language modelling (Chapter 3) and machine translation (MT) (Chapters 4 and 5). In this chapter we present our third and final model for the task of data-to-text generation. We present the Subword Segmental Pointer Generator (SSPG), an LSTM-based encoder-decoder that supplements subword segmental modelling with a copy mechanism. SSPG jointly learns to generate text describing structured data, to segment the text into subwords, and to directly copy entities from the data during text generation. To generate text with SSPG we propose a new decoding algorithm called unmixed decoding. We train SSPG on Triples-to-isiXhosa (T2X), an isiXhosa data-to-text dataset. SSPG outperforms established neural architectures for data-to-text, confirming our findings from previous chapters that subword segmental modelling proves beneficial in handling the complex morphology of isiXhosa. We also benchmark pretrained language models (PLMs) on T2X, some of which outperform SSPG but lag behind the unconventional, and overall best approach, of finetuning English $\rightarrow$ isiXhosa MT models. To go beyond reference-based metrics, we develop an evaluation framework that measures how accurately a text generation reflects the data content. This reveals that SSPG outperforms baselines because it combines subword modelling and copying, demonstrated by its improved ability to attach prefixes to copied entities. Finally, we repeat our experiments on Finnish data-to-text, replicating our findings for isiXhosa and reaffirming SSPG as a highly capable architecture for low-resource, agglutinative data-to-text generation.

Data triple:
(South Africa, leaderName, Cyril Ramaphosa)
([subject], [relation], [object])

(a) English text and template:
Cyril Ramaphosa is the leader of South Africa
[object] is the [relation][subject]

(b) isiXhosa text and template:
uCyril Ramaphosa yinkokheli yoMzantsi Afrika
u[object] yin[relation] yo[subject]

Figure 6.1: A translated example from the English WebNLG and Triples-to-isiXhosa (T2X) datasets that demonstrates the unique challenges presented by data-to-text for low-resource, agglutinative languages like isiXhosa. English data-to-text examples can be described by word-level templates, while T2X introduces the need for subword-based modelling.

6.1 Motivation

Data-to-text is the task of generating text that describes the content of structured data. For learning-based systems, the goal is to learn a mapping that transforms structured data (e.g. key-value pairs, triples, or entire tables) into natural language [46]. Training datasets consist of data instances matched with descriptive text passages. The type and complexity of structured data varies across datasets. For example, at the simpler end, the E2E dataset [102] consists of short restaurant reviews paired with key-value pairs containing review details (e.g. price range, cuisine type, customer rating). The WebNLG dataset achieves broader domain coverage [45], consisting of triples (subject, relation, object) extracted across several DBpedia categories, paired with text verbalising the triples. For more challenging data-to-text generation, the ROTOWIRE dataset [165] consists of a tables containing detailed information about basketball games, paired with articles summarising the games.

Data-to-text is a valuable natural language generation (NLG) task, as it enables the separate evaluation of the content of the text (*what to say*) and its style (*how to say it*) [165]. The majority of data-to-text datasets are in English or other high-resource languages, so such nuanced NLG evaluation is rarely conducted for low-resource languages.

The recent release[1] of the Triples-to-isiXhosa (T2X) [92] dataset has, for the first time, extended the data-to-text task to a Nguni language. T2X was constructed by manually translating part of the English WebNLG dataset [45]. It consists of triples (subject, relation, object) mapped to descriptive isiXhosa sentences (see Figure 6.1 for an example training instance). T2X is a challenging data-to-text dataset. It is not as large as the English WebNLG dataset, reflecting the challenge of working with the under-resourced Nguni language. Despite this data scarcity, to the best of our knowledge T2X is still the only Nguni-language dataset for an NLG task that is sufficiently large to train models from scratch (besides MT). It presents a valuable opportunity to benchmark the isiXhosa text generation capabilities of existing models and evaluate new architectures for the isiXhosa language.

Traditional approaches to data-to-text frame the task as a series of subtasks [124] and handle these subtasks separately through pipeline architectures [90]. More recent neural approaches cast data-to-text as a sequence-to-sequence task for end-to-end learning.[2] Vanilla encoder-decoder architectures and PLMs have proven highly effective [67, 101, 125], but data-to-text also presents an opportunity for modelling innovations. Data-to-text is a highly structured NLG task and there is room for exploiting this with task-informed inductive biases. The most common approach, often used as a strong baseline in data-to-text research, is the pointer generator [156]. This entails equipping the decoder of an encoder-decoder model with a copy mechanism – the ability to directly copy tokens from data during text generation [134]. A copy mechanism is a useful inductive bias for data-to-text, as demonstrated by the Figure 6.1 examples, in which some of the entities (e.g. “Cyril Ramaphosa”) are verbalised in the text exactly as they were represented in the data.

Besides pointer generators, several other encoder-decoder variants have been designed specifically for data-to-text (we refer to such models as *dedicated data-to-text architecures* throughout this chapter). Generally, these models supplement the standard LSTM-based

[1]This chapter is based on research we published as Meyer and Buys [92], which introduces the T2X dataset and the SSPG model. The second author (my doctoral supervisor) led the data collection, so I do not present the T2X dataset as a contribution of this thesis. As the lead author, my contributions were on the modelling side. I developed SSPG and conducted the T2X benchmarking, both of which are outlined in this chapter.

[2]Neural models have also been used as components in data-to-text pipelines [119, 120, 18], but in this chapter we view data-to-text as a sequence-to-sequence task for end-to-end learning, with no separable modules or intermediate output.

encoder-decoder architecture with additional structure to explicitly model some aspect of data-to-text generation that it can leverage to improve performance. One strategy is to model the underlying structure of the text, since data-to-text examples are often realisations of common templates (see Figure 6.1). Wiseman et al. [166] achieve this by inducing latent templates and generating text conditioned on these templates. Shen et al. [138] model the segmentation of text into fragments that are aligned with specific data records. Both models are LSTM-based encoder-decoder models that, in addition to modelling text structure, also use attention [8] to incorporate a pointer generator into their decoder [156].

Such dedicated architectures for data-to-text are designed with the high data availability and linguistic typology of English in mind. This is evident in that there are no studies on the role of subwords in data-to-text. Subwords are not essential for English data-to-text because English is morphologically simple – words are adequate units for modelling the complexity of datasets. Many examples in data-to-text datasets are instances of common templates. In English these are word-level templates (see Figure 6.1(a)). As a result, dedicated models for data-to-text do not apply subword segmentation, operating instead on word sequences [165, 166, 138]. This is not feasible for the Nguni languages due to their agglutinative morphology and conjunctive orthographies, since even simple templates are subword-based (see Figure 6.1(b)). The problem is compounded by the data scarcity of the Nguni languages – held-out test sets have high proportions of rare or new words, so subword modelling is essential. The problem of data-to-text for low-resource, agglutinative[3] languages therefore presents a new linguistic context for data-to-text that shifts modelling demands to subword-driven techniques.

In this chapter we present the subword segmental pointer generator (SSPG), a new neural model aimed at modelling data-to-text for low-resource, agglutinative languages like the Nguni languages. SSPG incorporates the copy mechanism of pointer generators into subword segmental modelling, thereby jointly learning subword segmentation, entity copying, and text generation. We achieve this by modifying the subword segmental machine translation

[3] In this chapter, our experiments only cover agglutinative languages that are conjunctively written (isiXhosa and Finnish), so for simplicity we use "agglutinative" to refer to the typical combination of agglutinative morphologies and conjunctive orthographies. We do not address languages that are disjunctively written.

architecture from Chapter 4 in two ways. Firstly, we change the encoder-decoder to be LSTM-based instead of Transformer-based, to match the tendency in established dedicated data-to-text architectures and enable training on much smaller datasets. Secondly, we equip the subword segmental decoder with a copy mechanism, allowing it to copy subwords directly from input data during text generation.

SSPG learns a subword segmentation scheme in conjunction with an entity copying strategy that optimises data-to-text performance. Crucially, combining subword modelling and copying allows SSPG to copy entities as subwords (for example, attaching prefixes to named entities to indicate noun classes in isiXhosa). We also propose a new decoding algorithm for generating text with SSPG, called unmixed decoding, since the dynamic decoding from Chapter 4 performs poorly with SSPG. SSPG adds to the line of work designing models trained from scratch for data-to-text. While PLMs are widely used, neural architectures equipped with inductive biases for data-to-text remain valuable for low-resource languages with few available high-quality pretrained options. SSPG successfully incorporates such an inductive bias, the copy mechanism, into our subword segmental paradigm.

6.2 Neural Architecture

SSPG is an encoder-decoder model, with a decoder that extends our subword segmental architecture from previous chapters with an attention-based copy mechanism. Following existing architectures for data-to-text [165, 166, 138], the encoder-decoder of SSPG is LSTM-based. LSTMs are well suited to such low-resource tasks and persist as the preferred neural architecture for training data-to-text models from scratch. In this section we outline how we incorporate a copy mechanism into the subword segmental architecture of SSPG, which is shown in Figure 6.2. We do not detail the training algorithm of SSPG since it is identical to that of SSMT, with the exception that subword probabilities computed in the dynamic programming algorithm incorporate probabilities from the copy mechanism. During training SSPG simultaneously learns (1) how to map data triples to text, (2) how to segment generated text into subwords, and (3) when to copy subwords directly from data during text generation.

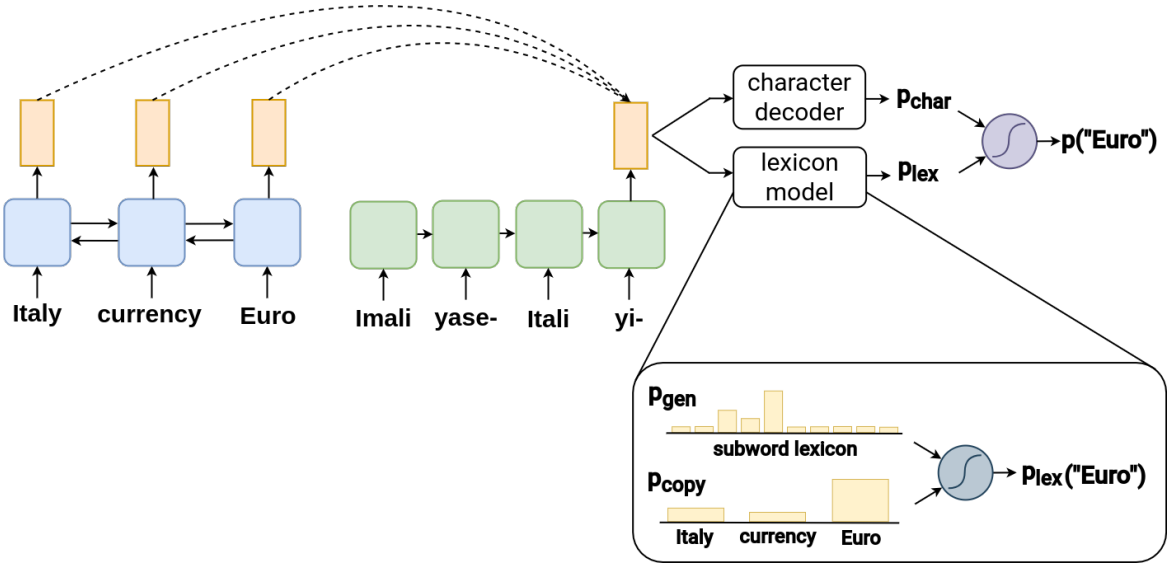

Figure 6.2: SSPG forward pass for (Italy, currency, Euro) → "Imali yase-Itali yi-Euro" ("The currency of Italy is the Euro"). The data triple is BPE-tokenised and encoded with a bi-LSTM. During generation, the preceding text is encoded with a character-level LSTM, and the next subword probability is computed as a mixture of a character-level decoder and a copy-equipped lexicon model (Equation 6.2).

6.2.1 BPE-based Data Encoder

The encoder is a standard neural encoder for data-to-text: a bi-LSTM that processes data as flattened sequences of byte-pair encoding (BPE) [135] tokens.[4] A BPE tokeniser is trained on the data portion of a data-to-text dataset, and applied to the data items before SSPG training. For example, the triple (Italy, currency, Euro) could be represented as the tokenised sequence "<s _Ita ly s> <r currency r> <o _Euro o>", where special tokens delimit the boundaries between subject, relation, and object. BPE is sufficient for data-side segmentation, since most data-to-text datasets in other languages (e.g. T2X and other translated versions of WebNLG [26]) have English data records.

Segmentations of named entities (such as "Italy" and "Euro") might not seem sensible, but having no data-side subword tokenisation resulted in much worse validation performance (we detail hyperparameter tuning in Section 6.4.3). Furthermore, many subject and object

[4]As in the case of SSMT, the SSPG encoder is not subword segmental. Our aim is to learn subword segmentation to optimise data-to-text generation, so we only adapt the text-generating decoder.

entities do contain non-named entities where subword segmentation could lead to useful segmentations e.g. the triple (Angola International Airport, runwayLength, 3800.0).

6.2.2 Subword Segmental Decoder

The decoder is *subword segmental*, i.e., it jointly models the generation and subword segmentation of the output text. We follow the dynamic programming algorithm for subword segmental sequence-to-sequence training outlined for SSMT in Chapter 4, but modify the Transformer-based model to be LSTM-based and extend it to copy subwords.

During training SSPG processes data-text pairs $(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ is a flattened triple of BPE tokens $\mathbf{x} = x_1, x_2, ..., x_{|\mathbf{x}|}$ and $\mathbf{y}$ is an unsegmented sequence of characters $\mathbf{y} = y_1, y_2, ..., y_{|\mathbf{y}|}$. We compute the probability of the output text conditioned on the input data as

$$p(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{s}:\pi(\mathbf{s})=\mathbf{y}} p(\mathbf{s}|\mathbf{x}), \tag{6.1}$$

where $\mathbf{s}$ is a sequence of subwords and π concatenates a sequence of subwords into its pre-segmented character sequence. Therefore we are marginalising over all possible subword segmentations of the output text $\mathbf{y}$. As for SSLM and SSMT, we constrain valid subword segmentations $\mathbf{s}$ to the segmentation of words into subword units. We prohibit subwords from crossing word boundaries and assume that word-separating characters (e.g. spaces and punctuation) are 1-character subword segments.

We model the probability of each subword sequence $\mathbf{s}$ with the chain rule, where each subword probability is computed with a mixture of a character-level decoder and a lexicon model as

$$p(s_i|\mathbf{y}_{<j}, \mathbf{x}) = g_j p_{\text{char}}(s_i|\mathbf{y}_{<j}, \mathbf{x}) + (1 - g_j) p_{\text{lex}}(s_i|\mathbf{y}_{<j}, \mathbf{x}), \tag{6.2}$$

where $\mathbf{y}_{<j}$ is the character sequence up to the character immediately preceding the next subword s_i. The character model p_{char} is a character-level LSTM and the lexicon model p_{lex} is a fully connected softmax layer over the subword lexicon, which consists of the top V (a

hyperparameter) most frequent character n-grams in training corpus. The coefficient g_j is computed by a fully connected sigmoid layer. As shown in Figure 6.2, decoder probabilities are conditioned on the input data and the text history by passing attention-based decoder output representations to the subword mixture model components.

As shown in Figure 6.2, the output text history is encoded with a character-level LSTM for computational tractability. We extract probabilities for all subsequent subwords up to a pre-specified maximum segment length and use dynamic programming to efficiently compute Equation 6.1, thereby summing over the probabilities of all possible subword segmentations of $\mathbf{y}$. In maximising Equation 6.1 during training, SSPG learns subword segmentation to optimise data-to-text generation.

6.2.3 Copying Segments

The model outlined so far jointly learns data-to-text generation and subword segmentation. This could be useful for low-resource, agglutinative languages, and we include it as a baseline called subword segmental decoder (SSD). However, it is missing an essential component of most data-to-text models: the ability to copy entities from data, as achieved by pointer generators. The pointer generator network is an encoder-decoder model equipped with a copy mechanism [156, 134]. During sequence-to-sequence training, this enables output sequence tokens to be copied directly from the input sequence.

Data-to-text requires the identification, extraction, and verbalisation of entities and attributes. Some types of entities (e.g. people, places, occupations) are verbalised exactly as they are stored in the structured data. The motivation behind pointer generators is to leverage this during training, allowing models to copy such entities directly from data where possible, instead of generating them from a fixed vocabulary. If models effectively learn when to make use of a copy mechanism (based on preceding context), they could generate previously unseen entities by through copying. This is why pointer generators have emerged as the standard approach to data-to-text. The ability to fill certain positions in the output text with entities copied from the input data is crucial to effective data-to-text modelling.

Therefore, we adapt subword segmental modelling for data-to-text through the addition of a subword-level copy mechanism.

We achieve this by including a conditional copy mechanism [52] in the lexicon model p_{lex}. We introduce a binary latent variable z_j at each character j, indicating whether the subsequent subword s_i is copied from data ($z_j = 1$) or generated from the lexicon ($z_j = 0$). We compute the p_{lex} in Equation 6.2 by marginalising out z_j as

$$p_{\text{lex}}(s_i|\mathbf{y}_{<j},\mathbf{x}) = p(z_j = 0|\mathbf{y}_{<j},\mathbf{x})p_{\text{gen}}(s_i|\mathbf{y}_{<j},\mathbf{x}) + p(z_j = 1|\mathbf{y}_{<j},\mathbf{x})p_{\text{copy}}(s_i|\mathbf{y}_{<j},\mathbf{x}), \quad (6.3)$$

where p_{gen} is a softmax layer over the lexicon and p_{copy} is the attention distribution over the data tokens. The probabilities $p(z_j = 0|\mathbf{y}_{<j},\mathbf{x})$ and $p(z_j = 1|\mathbf{y}_{<j},\mathbf{x})$ can be viewed as mixture model coefficients (similar to g_j in Equation 6.2). They are computed by a fully connected sigmoid layer, allowing SSPG to learn (based on context) when it can rely on the lexicon's generation model and when it should copy subwords directly from data input.

6.3 Unmixed Decoding

Tokenisation-based pointer generator networks generate text by performing beam search over the joint distribution of the subword vocabulary and input tokens. This is not directly applicable to SSPG, as for other subword segmental models, because the components of the mixture model (Equation 6.2), operate at two levels: character and subword probabilities. In Chapter 4 we address this feature of subword segmental modelling by introducing dynamic decoding, which we use to generate translations with SSMT. Dynamic decoding combines information from both mixture components, generating one character at a time. During development, we found that SSPG and dynamic decoding failed to outperform tokenisation-based pointer generators on validation data (we discuss this in more detail in Section 6.5.2, our ablation study). SSPG validation performance improved drastically when we developed a new decoding algorithm, which we call *unmixed decoding* (see Figure 6.3).

We first describe the greedy version of the algorithm. SSPG subword probabilities are a mixture of three distributions: the character decoder, lexicon model, and copy mechanism.

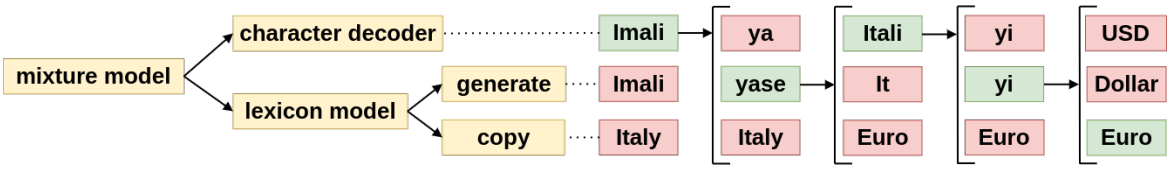

Figure 6.3: Unmixed decoding generating the isiXhosa text "Imali yaseItali yiEuro" ("The currency of Italy is the Euro") based on the data triple (Italy, currency, Euro). During each decoding step, the algorithm compares probabilities from three components (character model, lexicon generation model, and lexicon copy mechanism). The subword with the highest probability from any of these components is selected for generation, ensuring efficient generation and effective use of the learned mixture model for subword prediction.

Unmixed decoding extracts next-subword probabilities from the three distributions *separately* and selects the next subword with the highest probability across the three distributions, each considered independently. At each decoding step we compute the top next-subword probability from each distribution respectively as

$$p^*_{\text{char}} = \max_s g p_{\text{char}}(s|\cdot),$$
$$p^*_{\text{gen}} = \max_s (1-g) p(z=0|\cdot) p_{\text{gen}}(s|\cdot),$$
$$p^*_{\text{copy}} = \max_s (1-g) p(z=1|\cdot) p_{\text{copy}}(s|\cdot),$$

where we omit the conditioning variables of Equations 6.2 and 6.3 for simplification. We store the candidate subwords s^*_{char}, s^*_{gen}, s^*_{copy} corresponding to these probabilities. From these candidates, we then select the subword s^* corresponding to the highest probability among p^*_{char}, p^*_{gen}, p^*_{copy}, so the next subword generated is

$$s^* = \arg\max_s \{p^*_{\text{char}}, p^*_{\text{gen}}, p^*_{\text{copy}}\}. \tag{6.4}$$

This process is repeated until the next subword is the end-of-sequence token. It is straightforward to combine unmixed decoding with beam search by extracting the top k subword candidates from each mixture component (resulting in $3k$ initial candidates), ranking these probabilities, and continuing with the top k subwords.

Each subword generated by unmixed decoding is proposed by one of the mixture components (as shown in Figure 6.3). During training, SSPG learns in which contexts to use the character decoder, generate from the lexicon, or copy tokens from data. Unmixed decoding leverages this information during generation. For example, when the model is "confident" that the next subword should be copied from data, p^*_{copy} will be greater than p^*_{char} and p^*_{gen}, so the next subword is generated by the copy mechanism.

Unmixed decoding assumes that each subword is generated by *one* of the mixture model components – either it is generated one character at a time by the character model, it is generated from the subword lexicon, or it is copied directly from the data input. While dynamic decoding generates one character at a time and combines probabilities from all mixture components, unmixed decoding generates entire subwords based on information from individual mixture components. In that sense it is a greedier decoding algorithm than dynamic decoding, which we believe contributes to its success in this context. The T2X dataset is too small for dynamic decoding to work well – some mixture components have high prediction uncertainty due to being under-trained, so attempting to incorporate information from them only introduces noise.

6.4 Experimental Setup

We implement SSPG as a sequence-to-sequence model in the fairseq library[5] and publicly release our code.[6] We now describe the main dataset for our isiXhosa experiments, T2X, introduce the data-to-text baselines that we benchmark on T2X alongside SSPG, and discuss our hyperparameter tuning and training procedures.

6.4.1 T2X Dataset

WebNLG [45] is an English data-to-text dataset consisting of RDF triples from DBpedia paired with text verbalising the triples. Each example is one or more triples (up to seven)

[5] https://github.com/facebookresearch/fairseq
[6] https://github.com/francois-meyer/sspg

	Train	Valid	Test
WebNLG 1-triples	3 114	392	388
T2X triples	2 413	391	378
T2X verbalisations	3 859	600	888

Table 6.1: T2X dataset statistics

paired with a crowd-sourced verbalisation of one or more sentences. Multiple verbalisations are included for a large portion of the examples. The dataset has been expanded and translated into Russian, using machine translation and manual post-editing [17]. Recently translations have been released for Maltese, Irish, Breton, and Welsh.[7] These datasets cover smaller subsets of WebNLG with up to 1,665 examples per language.

Triples-to-isiXhosa (T2X) [92] was constructed by manually translating a subset of English WebNLG to isiXhosa. It is based on the 1-triples in WebNLG version 2.1.[8] While the dataset only includes examples with single triples, it spans 15 DBPedia categories to achieve broad domain coverage. Three categories (Astronaut, Athlete and WrittenWork) are not included in the training data, only in the validation and test sets. Dataset statistics are listed in Table 6.1 and example data-text pairs are provided in Figure 6.1 and Tables 6.2 & 6.6.[9] In the training and validation sets, only one isiXhosa verbalisation is provided per triple for most domains, while the test set has multiple verbalisations (up to three) for most examples. Multiple verbalisations correspond to different ways of describing the same data triple, capturing variations in phrasing for more nuanced evaluation. Equivalent verbalisations usually contain synonyms or different word orderings, as shown in the Table 6.2 example.

T2X maps single triples to isiXhosa verbalisations. Unlike WebNLG, it does not contain examples with multiple triples. In that sense it is a simpler task, but in other ways T2X is more challenging than existing datasets. For example, E2E [102] covers one domain (restaurants) and its examples follow a limited set of templates (e.g "[RESTAURANT NAME] is a [TYPE] restaurant in [AREA]"). T2X covers 15 domains and 286 relation types. It would be difficult

[7] https://github.com/WebNLG/2023-Challenge
[8] https://huggingface.co/datasets/web_nlg
[9] The full T2X dataset is public available at https://github.com/francois-meyer/t2x.

Data	(Germany, leaderName, Angela Merkel)
Text #1	Inkokeli yaseJamani ngu-Angela Merkel. *The leader of Germany is Angela Merkel.*
Text #2	U-Angela Merkel yinkokeli yaseJamani. *Angela Merkel is the leader of Germany.*

Table 6.2: An example T2X data triple mapped to two isiXhosa verbalisations (*with English translations*).

to model T2X with a template-based approach, since it requires some degree of generalisation and fluency, which is why learning-based approaches are more suitable.

T2X poses a different type of modelling challenge than English data-to-text datasets, because of the highly agglutinative morphological system of the isiXhosa language. As shown in Figure 6.1, the underlying syntactic schemas for isiXhosa data-to-text generations are inherently subword-based, with morphemes being the primary units of meaning. For English, a word-based model would cover most examples. For isiXhosa, a subword-based model is essential for even minimal text generation.

6.4.2 Baselines

We benchmark several established data-to-text approaches on T2X to compare against SSPG. Among our baselines, three are trained from scratch (PG, NT, SSD) and nine are pretrained models (either text-to-text PLMs or English $\rightarrow$ isiXhosa MT models) finetuned on T2X.

Trained from Scratch (Dedicated Data-to-Text Architectures)

- **Pointer generator (PG)** [156] is an LSTM-based encoder-decoder model with a copy mechanism. This is commonly employed as a data-to-text baseline [165, 68].

- **Neural templates (NT)** [166] learn latent templates for text. The LSTM-based decoder uses a hidden semi-markov model to jointly model templates and text generation. They show that the resulting templates sometimes correspond to common field types such as names, dates, and occupations.

- **Subword segmental decoder (SSD)** is an LSTM-based sequence-to-sequence model with a subword segmental decoder. It is equivalent to SSPG without a copy mechanism. We use dynamic decoding during generation.

Pretrained Models (Finetuned for Data-to-Text)

The majority of our pretrained baselines are text-to-text PLMs based on T5, which have proven highly effective for structured NLG tasks like data-to-text [67, 101, 125] and still outperform decoder-only LLMs on low-resource sequence-to-sequence tasks [104]. We finetune several T5 variants, some of which have been adapted for isiXhosa and other Nguni languages. To further explore the potential of transfer learning for T2X, we turn to the only other publicly available encoder-decoder models that are trained on isiXhosa: machine translation (MT) models.

- **mT5-base & mT5-large** [169] are multilingual versions of T5 [123], covering 101 languages, including isiXhosa and isiZulu.
- **Afri-mT5-base** [1] adapts mT5-base for 17 African language, including isiXhosa and isiZulu, through continued pretraining.
- **ByT5-base & ByT5-large** [168] are text-to-text PLMs for 101 languages, including isiXhosa and isiZulu. ByT5 is trained with a span denoising objective [123], but processes text as UTF-8 bytes instead of subword tokens. Training on bytes has shown potential as an alternative to subword tokenisation for low-resource languages [1, 35].
- **Afri-ByT5-base** [1] adapts ByT5-base for 17 African languages, including isiXhosa and isiZulu, through continued pretraining.
- **Nguni-ByT5-large** [93] adapts ByT5-large for the four Nguni languages (isiXhosa, isiZulu, isiNdebele, and Siswati) through continued pretraining.[10]

[10]During my PhD, I completed a research internship at the National Institute of Information and Communications Technology (NICT) in Kyoto, Japan. We developed Nguni-ByT5-large as part of the research conducted for this internship, in an effort to adapt and benchmark PLMs for the Nguni languages. It is not directly related to the main topic of this thesis, subword segmental modelling, so we do not present it as a thesis contribution. Instead, we employ it it as an additional PLM baseline to benchmark against SSPG.

Model	lr	dropout	batch	vocab	other
PG	1e-3	0.3	4	500	
NT	0.5	0.3	4	500	states: 10, max segment length: 20
SSD	1e-3	0.3	4	250	lexicon: 250, max segment length: 5
SSPG	1e-3	0.5	4	1k	lexicon: 1k, max segment length: 5
mT5 variants	1e-4	0.1	8	250k	warmup: 0, lr schedule: fixed
ByT5 variants	1e-4	0.1	8	250k	warmup: 0, lr schedule: fixed
BPE MT	1e-4	0.3	16	10k	warmup: 500, lr schedule: inverse sqrt
SSMT	1e-4	0.3	16	5k	warmup: 0, lr schedule: fixed

Table 6.3: Hyperparameter settings for our final T2X models, chosen based on validation chrF++ scores. Some hyperparameters are the same for all our models trained from scratch, including the number of LSTM layers in the encoder & decoder (1) and the size of the embedding & hidden layers (128).

- **Bilingual pretrained MT (BPE MT & SSMT)** We finetune two bilingual MT models trained in Chapter 4 for English $\rightarrow$ isiXhosa: a BPE tokenisation-based based model and our SSMT model. Low-resource languages like isiXhosa are severely underrepresented in massively multilingual pretraining. In contrast, isiXhosa makes up 100% of the target-side pretraining corpus of our MT models. We expect the isiXhosa translation generation capabilities to transfer to isiXhosa data-to-text generation.

6.4.3 Hyperparameters

For each model we perform a hyperparameter grid search. We select the final model based on validation chrF++ score, since it measures the subword-level performance so critical to T2X [116, 9]. The final hyperparameter values are listed in Table 6.3. Some of the hyperparameters are common to all models, while others are unique to specific models. For NT we tune the number of discrete states and the maximum length of segments. For SSD and SSPG we tune the maximum segment length and the lexicon size. For the pretrained models we experimented with different learning rate regimes for finetuning.

We put substantial effort into tuning the subword tokenisation schemes of our baselines, given the importance of subword modelling for T2X. Prior work trained dedicated data-to-

text models without subword tokenisation [165, 166]. However, we found that subword tokenisation, using either BPE [135] or unigram language model (ULM) [81], improves validation performance for all our baselines trained from scratch. BPE and ULM performed comparably during development in validation-based evaluations, so we employed BPE for baselines in subsequent experiments. For NT and PG we construct a shared BPE vocabulary, applied to the joint data and text corpora. For SSD and SSPG we train BPE on the data-side (the flattened triples) of the data-to-text dataset, since we only pre-segment the data-side for these models. Applying no subword tokenisation to the data triples results in notably worse validation performance for SSD and SSPG. We tune the BPE vocabulary size separately for each baseline over the range [250, 500, 1k, 2k, 5k] (see Table 6.3 for final subword vocabulary sizes). We cannot tune the subword tokenisation of our pretrained baselines, as they are equipped with pretrained subword tokenisers.

6.5 Experiment 1: IsiXhosa Data-to-Text

6.5.1 Reference-based Metrics

We compute several automatic metrics to measure different aspects of overlap between generations and reference texts: BLEU [109], chrF [115], chrF++ [116], NIST [30], METEOR [84], ROUGE [87], and CIDEr [155]. The results for T2X are reported in Table 6.4.

SSPG outperforms the other dedicated data-to-text models trained from scratch across all metrics. We attribute the low scores of NT to the fact that BPE tokens are not ideal units for learning templates (NT is intended to learn word-level templates, but this led to even worse performance on T2X). Considering that PG outperforms SSD, the impact of learning subword segmentation appears secondary to copying. SSPG combines both features to achieve substantial performance gains over PG, with increases of 2.21 on chrF++ and and 1.11 on BLUE. This establishes SSPG as a highly capable dedicated neural architecture for agglutinative data-to-text generation.

SSPG outperforms mT5 and ByT5 (base and large variants) on chrF++ and BLEU. The fact that SSPG, trained from scratch, outperforms these text-to-text PLMs highlights the un-

	Model	chrF++	chrF	BLEU	NIST	MET	ROU	CID
From scratch	PG	46.24	51.09	18.90	4.32	19.84	37.48	1.32
	NT	38.00	42.28	12.02	3.43	15.97	27.00	0.85
	SSD	44.77	49.91	16.16	4.14	19.85	34.97	1.16
	SSPG	**48.45**	**53.46**	**20.01**	**4.51**	**21.92**	**38.68**	**1.38**
Finetuned PLMs	mT5-base	41.45	46.40	14.32	3.87	20.66	33.75	1.06
	Afri-mT5-base	42.42	47.64	16.11	4.02	20.49	34.10	1.15
	mT5-large	46.92	52.05	19.87	4.63	23.13	39.26	1.41
	ByT5-base	46.17	51.16	17.62	4.32	22.60	35.69	1.21
	Afri-ByT5-base	51.95	56.67	22.56	5.21	26.11	44.67	1.64
	ByT5-large	44.16	48.87	15.95	4.08	21.92	34.25	1.13
	Nguni-ByT5-large	**52.96**	**57.71**	**24.56**	**5.34**	**26.61**	**45.09**	**1.71**
Finetuned MT	BPE MT	**56.05**	**61.53**	**27.56**	**5.62**	**27.24**	**47.49**	**1.88**
	SSMT	54.01	59.44	24.19	5.33	26.13	44.34	1.61

Table 6.4: T2X test results. Best scores per category are **bold** and best scores overall are underlined.

reliability of massively multilingual pretraining for low-resource text generation. This can be attributed to the fact that isiXhosa makes up a very small proportion of the pretraining corpora of mT5 and ByT5. We only observe gains from pretraining with PLMs specifically adapted for African languages. Both Afri-ByT5-base (adapted for 17 African languages including isiXhosa) and Nguni-ByT5-large (adapted for the four Nguni languages) outperform SSPG. These results align with the prevailing knowledge [6, 1, 66] that continued pretraining on a more limited set of languages can greatly improve the downstream performance of PLMs for low-resource languages. Interestingly, SSPG is not outperformed by Afri-mT5-base, which shows the advantage of byte-based processing. Once again, subword tokenisation is suboptimal for low-resource, agglutinative text generation, as demonstrated by the superior performance of SSPG and byte-level PLMs over tokenisation-based PLMs.

We see further gains from pretraining when we turn to the unconventional approach of finetuning MT models. Overall, the best model benchmarked on T2X is the finetuned English $\rightarrow$ isiXhosa BPE MT model. Finetuning English $\rightarrow$ isiXhosa SSMT achieves the second best results overall. SSMT can adjust its subword segmentations during finetuning, but this

Model and decoding algorithm	chrF++	BLEU
BPE +copy +beam search (PG)	48.25	18.81
Subword segmental models		
+copy +unmixed decoding (SSPG)	**49.41**	**20.35**
+copy +dynamic decoding	43.21	14.16
–copy +unmixed decoding	47.11	16.87
–copy +dynamic decoding (SSD)	46.84	17.54

Table 6.5: T2X validation scores for ablations.

does not improve finetuning performance over the fixed tokenisation of the BPE-based MT model. This aligns with our findings on cross-lingual finetuning in the previous chapter (Section 5.4.2), where BPE outperformed SSMT in most finetuning instances (SSMT only improved finetuning in cases where models were adapted for languages with vastly different morphologies). BPE MT outperforming SSMT shows that learning subword segmentation for T2X is only advantageous when paired with a subword copy mechanism (our upcoming ablation study confirms this for models trained from scratch as well).

The fact that finetuned MT models outperform PLMs is surprising, and to our knowledge, has not been explored in prior data-to-text work. This shows that pretraining and finetuning has potential for this task, but underlines the fact that standard pretrained solutions (text-to-text PLMs) still have significant limitations for isiXhosa. Based on our results, finetuning a pretrained MT model can be an effective approach to data-to-text modelling for certain languages. For high-resource languages finetuning text-to-text PLMs might yield better results, but for many low-resource languages MT models will be the best pretrained option available. For extremely low-resource languages without high-quality MT models, SSPG is the best choice for training a data-to-text model from scratch on agglutinative morphologies.

6.5.2 Ablation Study

Table 6.5.2 shows the effect of different SSPG components on performance. A subword segmental encoder-decoder without a copy mechanism (SSD) falls well short of a BPE-based pointer generator (PG). Adding the copy mechanism to subword segmental modelling

improves performance, but only if we use unmixed decoding, which is crucial for leveraging the copying ability of SSPG.

Dynamic decoding leads to worse performance in all instances. Adding a copy mechanism to the subword segmental mixture model could render the other mixture components (the lexicon model and character-level decoder) less reliable in certain contexts. At every generation step, and independent of context, dynamic decoding incorporates information from all mixture components. Unmixed decoding only relies on a mixture component in contexts where the component assigns a high probability mass to a specific prediction, discarding information from components with high entropy distributions (high uncertainty in their predictions). This distinction is especially important for smaller datasets like T2X, where limited training data leads to less reliable predictions across components. In such cases, dynamic decoding spreads its reliance too broadly, while unmixed decoding mitigates the risks associated with high-entropy predictions by focussing on the most confident predictions. By adopting a more "greedy" approach, unmixed decoding proves advantageous in low-resource scenarios. While dynamic decoding is useful for high-resource tasks like MT, unmixed decoding is better suited for generating text with subword segmental models trained on smaller datasets.

6.6 Experiment 2: Content-based Analysis

Data-to-text presents an opportunity for more interpretable evaluation, such as quantifying how accurately generations reflect data content. To achieve this we adapt the extractive evaluation framework of Wiseman et al. [165] for T2X.

6.6.1 Extractive Evaluation Framework

In T2X (as in other WebNLG translations) the triples are in English, while the text descriptions are in isiXhosa. This adds a layer of complication to modelling the dataset. Some entities can be directly copied from data (they are the same in English and isiXhosa), but others should be translated to isiXhosa. For example, in Figure 6.1 the name "Cyril Ramaphosa" should be

Data	(a) (**South Africa**, capital, *Cape Town*)	(b) (**Christian Panucci**, club, *Inter Milan*)
Ref	Ikomkhulu lo**Mzantsi Afrika** li*Kapa*.	U**Christian Panucci** udlalela i-*Inter Milan*.
SSPG	I-*Cape Town* likomkhulu lase**South Africa**.	U**Christian Panucci** udlalela i-*Inter Milan*.
PG	U*Cape Town* likomkhulu lase-**Afrika**.	U**Christian Puucci** udlalela i-*Indter Milan*.
BPEMT	Ikomkhulu lo**Mzantsi Afrika** yi*Kapa*.	U**Christian Panucci** udlalela i-*Inter Milan*.
Data	(c) (**Ethiopia**, leaderName, *Mulatu Teshome*)	(d) (**Canada**, language, *English*)
Ref #1	U*Mulatu Teshome* yinkokheli yase-**Ethiopia**.	Isi*Ngesi* lulwimi oluthethwa e**Khanada**.
Ref #2	Igama lenkokheli e-**Ethiopia** ngu*Mulatu Teshome*.	Ulwimi lwesi*Ngesi* luthethwa e**Khanada**.
SSPG	U*Mulatu Teshome* yinkokeli yase-**Ethiopia**.	e**Canada** kuthetwa isi*Ngesi*.
PG	Inkokeli yase-**Ethiopia** ngu*Mulatu Teshome*.	Ulwimi lwesi*Ngesi* luthethwa e**Canada**.
BPEMT	U*Mulatu Teshome* yinkokeli yase-**Ethiopia**.	Isi*Ngesi* lulwimi oluthethwayo e**Canada**.

Table 6.6: Examples from T2X with model outputs. Subject verbalisations are **bold** and object verbalisations *italicised* to show that some entities should be copied directly while others should be translated. Green and red show correctly and incorrectly generated entities according to our evaluation framework.

copied, but the country "South Africa" should be translated to "Mzantsi Afrika". Additionally, named entities in isiXhosa are usually prefixed, e.g. "yase-Ethiopia" or "liKapa", in which case prefixal morphemes should be appended to copied entities. Our aim is to assess whether models are implementing the copying and translation of entities correctly.

In more complex data-to-text datasets, not all entities in the data should be mentioned in the text – only those that are important to the summary. T2X is simpler. Since each example consists of a single triple, we can be sure that each text should mention the subject and object in the data. To correctly verbalise data entities in T2X, models have to learn when to copy entities and when to generate isiXhosa translations of entities. Some models *overcopy*, including English words in the output text where translations are required (see the example SSPG generation in Table 6.6(a)). Other models copy inaccurately, resulting in missing or partially copied entities in the text (see the example PG generation in Table 6.6(b)). We can quantify this by casting the copying subtask as an information retrieval problem: does a generation contain the correctly copied entities?

For each example we check if the subject/object should be directly copied to the output text. We do this by checking if the entity string from the data triple is present in the reference text. If it is, the entity should be directly copied (like "Ethiopia" in Table 6.6(c)). If it is not,

		Subject			Object			Rel
	Model	P	R	F1	P	R	F1	acc
From scratch	PG	71.30	63.37	67.10	**77.39**	74.40	75.86	**75.38**
	NT	72.65	73.25	72.95	67.14	68.12	67.63	38.14
	SSD	**76.01**	84.77	80.16	71.26	59.90	65.09	67.27
	SSPG	74.83	**88.07**	**80.91**	75.42	**85.99**	**80.36**	70.57
Finetuned PLMs	mT5-base	70.27	85.60	77.18	73.79	73.43	73.61	53.45
	Afri-mT5-base	70.90	86.18	77.80	74.07	75.47	74.77	65.01
	mT5-large	70.78	89.71	79.13	75.22	82.13	78.52	69.37
	ByT5-base	71.14	86.18	77.94	72.06	92.45	80.99	63.27
	Afri-ByT5-base	68.53	94.72	79.52	73.78	92.92	82.25	72.30
	ByT5-large	67.91	88.62	76.90	72.47	84.43	78.00	46.64
	Nguni-ByT5-large	**74.45**	**95.93**	**83.84**	**75.29**	**93.40**	**83.37**	**79.59**
Finetuned MT	BPE MT	74.81	83.13	78.75	77.09	**84.54**	80.65	**87.69**
	SSMT	**77.78**	**86.42**	**81.87**	**83.17**	83.57	**83.37**	83.78

Table 6.7: T2X extractive results. Best scores per category are **bold** and best overall are underlined.

the entity should be translated (like "South Africa" in Table 6.6(a)). We do not extract the translated entity from the reference text to check if it is generated correctly – the example is simply labelled as an instance where copying should not occur. We do not evaluate entity translation capabilities. We are only interested in evaluating whether models are able to master the subtask of predicting whether or not an entity should be copied. Each data-to-text example contains a binary decision for the subject and object, i.e., to translate or to copy. Our evaluation framework tests how well models learn this decision.

Relation prediction We cannot apply extraction to relations because, unlike subjects and objects, they can never be copied untranslated from data. We follow Wiseman et al. [165] in training a relation prediction model which we use to estimate how well models capture relations. We finetune AfroXLMR-large [6], a masked language model adapted for 17 African languages including isiXhosa, to predict T2X relation types based on reference texts. The model achieves 85% heldout accuracy, which is high enough for estimating relation verbalisation capabilities. We apply this predictor to test set generations of models, producing

(b)	Ref	UWilliam M.O. Dawson wazalelwa...
	PG	IWilliam M. O. Dawson wazalelwa...
(a)	Ref	UNorbert Lammert yinkokeli yaseJamani.
	PG	UNorbu Lammert yinkokeli yaseJamani.
(c)	Ref	I-Dublin yinxalenye yeLeinster.
	PG	IDubler yinxalenye yeLeinster.

Table 6.8: PG output compared to reference texts. Red shows where PG fails on subword copying.

a predicted relation for each example. We compare these to the correct relations in the test set data to estimate how accurately models describe relations.

6.6.2 Results

Table 6.7 contains the results of our extractive evaluation, revealing a more mixed account of model performance than automatic metrics, based on the data content of model generations. Among the models trained from scratch, SSPG achieves the highest F1 scores. Its precision is lower than its recall, indicating some overcopying, but not to such an extent that it undermines its F1 scores. Comparing precision and recall across models suggests that SSPG balances copying and translating better than the other models trained from scratch – it learns in which contexts to copy directly and when to translate instead.

PG outperforms SSPG on relations, which is based on descriptive isiXhosa text (e.g. "yinkokheli" in Fig. 6.1 means "is the leader"). The BPE subwords of PG are sufficient for modelling these isiXhosa phrases. However, PG struggles with the subword modelling of entities. isiXhosa has many prefixes that indicate grammatical roles (e.g. "*u*John" indicates singular personal proper noun). T2X requires attaching these prefixes to entities correctly. A qualitative analysis of generations reveals that PG struggles with this (see Table 6.8). PG generates entities correctly when they can be copied without alteration, but struggles when they have to be modified according to morphological rules. For example, PG sometimes attaches the incorrect prefix (Table 6.8 (a)). In other cases, attaching the prefix leads to errors in copying the root entity (Table 6.8 (b), (c) and Table 6.6 (b)). SSPG does not seem to make

these types of mistakes. By jointly modelling subword segmentation and copying, SSPG learns to combine the two when required. The subword tokenisation of SSPG fails to do so reliably.

As in the automatic metrics, SSPG outperforms mT5 but is outperformed by Nguni-ByT5-large and the MT models. SSMT achieves high F1 scores for subjects and objects, while BPE MT achieves the highest relation accuracy. This reiterates our findings for the models trained from scratch: BPE subwords are sufficient for descriptive isiXhosa phrases, but modelling subword segmentaion allows SSMT to model subword-based changes to entities.

6.7 Experiment 3: Finnish Data-to-Text

To see if our findings generalise to another agglutinative language we perform experiments on Finnish data-to-text (to the best of our knowledge Finnish is the only other agglutinative language with a data-to-text dataset). Our experimental setup and baselines are the same used in our isiXhosa experiments, except that we do not evaluate the PLMs adapted for African languages (Afri-mT5, Afri-byT5 and Nguni-byT5). The PLMs that we do finetune, mT5 and ByT5, do include Finnish in their pretraining corpora. For our finetuned bilingual MT models, we use the English $\rightarrow$ Finnish models trained for Chapter 4.

6.7.1 Dataset

The Finnish Hockey dataset [68] is based on news articles about ice hockey games. It contains game statistics and text spans describing corresponding game events. The data records are more complex than T2X (up to 12 records, depending on event type), but the texts are still single sentences. Our models are trained on the 6,159 data records that are aligned with single text spans. We use the data extraction and preprocessing scripts of Kanerva et al. [68], including their tokenisation and detokenisation strategies for evaluation.

	Model	chrF++	chrF	BLEU	NIST	MET	ROUGE	CID
From scratch	PG	37.57	37.98	19.24	4.54	22.74	43.41	2.10
	NT	32.13	33.70	11.95	3.64	19.17	36.18	1.45
	SSD	33.68	33.78	17.03	4.17	21.53	41.48	1.88
	SSPG	**40.12**	**41.00**	**20.87**	**4.63**	**23.97**	**44.52**	**2.13**
Finetuned PLMs	mT5-base	28.18	30.53	8.39	2.71	14.56	28.49	1.10
	mT5-large	32.70	34.22	13.40	3.46	19.48	39.93	1.75
	ByT5-base	**36.77**	**37.69**	**21.16**	**4.74**	**23.43**	**44.91**	**2.23**
	ByT5-large	29.44	31.17	9.23	3.24	15.31	29.53	1.14
Finetuned MT	BPE MT	**42.37**	**41.59**	**22.36**	**5.04**	**24.87**	**46.04**	**2.25**
	SSMT	36.59	38.52	15.54	4.03	21.62	38.99	1.62

Table 6.9: Finnish test results. Best scores per category are **bold** and best scores overall are underlined.

6.7.2 Results

Table 6.9 reports the automatic metrics achieved by our models on the test set. Our results for PG are very close to those reported for a pointer generator model by Kanerva et al. [68]. The relative performance of models is similar to T2X. SSPG is the best model trained from scratch, while the finetuned English $\rightarrow$ Finnish BPE MT model is best overall. Both models surpass the benchmark set by Kanerva et al. [68], with SSPG increasing the BLEU score by 1.2 and BPE MT by 2.69. Among the finetuned PLMs, the best performing model is ByT5-base, reaffirming the robustness of byte-level PLMs for agglutinative languages. As for T2X, the smaller base variant of ByT5 outperforms the large variant, while the reverse is true for mT5. This aligns with the findings of Xue et al. [168], where the small and base variants of ByT5 were shown to be particularly effective. Byte-level models require very few vocabulary parameters, so smaller variants can dedicate a much greater proportion of parameters to dense connections that learn data-to-text generation.

Table 6.10 presents examples of model generations from the Finnish Hockey test set. The first two examples demonstrate that, as we found for T2X, SSPG sometimes copies data values more accurately than tokenisation-based models.

Example (a): BPE MT incorrectly copies the time.

Data	length: short, type: penalty, team: JYP **guest**, player: Jyrki Välivaara, minutes: 2, exact_time: 57.12, approx_time: 44, period: 3
Reference	JYPin Jyrki Välivaara otti kakkosen ajassa 57.12 .
SSPG	JYPin Jyrki Välivaara karkasi jäähyaitioon ajassa 57.12 .
PG	JYP : n Jyrki Välivaara otti jäähyn ajassa 57.12 .
BPE MT	JYP : n Jyrki Välivaara lähetettiin jäähyaitioon ajassa 51.27 .

Example (b): PG incorrectly copies the time.

Data	length: short, type: goal, team: HIFK home, player: Mikko Kousa, assist: Jerry Ahtola, Mikael Granlund, team_score: 4, score: 4 – 2, exact_time: 35.33, approx_time: 44, period: 2, goaltype: increase_lead_goal, abbrevs: yv
Reference	HIFK meni 4 – 2 - johtoon ajassa 35.33 .
SSPG	HIFK meni 4–2-johtoon ajassa 35.33 .
PG	Mikko Kousa onnistui maalinteossa ajassa 45.33 .
BPE MT	HIFK : n Mikko Kousa teki 4-2 ylivoimamaalin .

Example (c): BPE MT matches the reference text almost exactly.

Data	length: short, type: save, team: HIFK guest, player: Jan Lundell, saves: 31
Reference	Jan Lundell HIFK : n maalissa torjui 31 kertaa .
SSPG	HIFK : n maalivahti Jan Lundell torjui 31 kiekkoa .
PG	HIFK : n maalivahti Jan Lundell ylsi 31 torjuntaan .
BPE MT	Jan Lundell HIFK : n maalissa torjui 31 kiekkoa .

Example (d): SSPG correctly selects the relevant data fields for its generation.

Data	length: short, type: goal, team: Ilves home, player: Cory Murphy, assist: Raimo Helminen, Vesa Viitakoski, team_score: 4, score: 4 – 3, exact_time: 60.48, approx_time: 14, period: 4, goaltype: final_goal, deciding_goal, leading_goal
Reference	Cory Murphy teki voittomaalin jatkoajalla ajassa 60.48 .
SSPG	Lappeenrantalaisten Cory Murphy teki voittomaalin ajassa 60.48 .
PG	Voittomaalin viimeisteli Cory Murphy .
BPE MT	Voittomaalin teki Cory Murphy jatkoerässä Raimo Helmisen syötöstä .

Table 6.10: Examples of model generations from the Finnish Hockey test set, with correctly generated word forms highlighted in green.

6.8 Limitations

SSPG takes longer to train than PG because of the additional computation required by its dynamic programming algorithm for summing over latent subword segmentations. The increase in training time depends on the model's maximum segment length. Our final SSPG model has a maximum segment length of 5 characters and took aprroximately 4 hours to train on a single NVIDIA A100, as opposed to the 20 minutes training required for our final PG model. Our unmixed decoding algorithm is computationally more efficient than dynamic decoding, but 3 times slower than standard beam search, since it operates over combined vocabulary of 3 separate models (the character, lexicon, and copy models).

Our experiments are limited to two datasets for two languages. To the best of our knowledge, no data-to-text datasets are available for other agglutinative languages. We cannot make claims about how well SSPG will generalise to other languages and differently structured data-to-text tasks. Our results are very similar across isiXhosa and Finnish and the differences in performance between models are substantial enough that we can confidently claim some generalisability, but only in a narrow linguistic context (simple data-to-text datasets for low-resource agglutinative languages).

6.9 Conclusion

In this chapter we presented SSPG, our third and final subword segmental model aimed at data-to-text for low-resource, agglutinative languages. The neural architecture of SSPG combines our subword segmental architecture with a copy mechanism for copying data entities during generation. We train SSPG and three existing dedicated data-to-text models from scratch on T2X. SSPG outperforms these tokenisation-based baselines, improving chrF++ [116] by 2.12 and BLEU by 1.05. This shows that conventional approaches to data-to-text (e.g. pointer generators and neural template models) are not well suited to the unique challenges posed by low-resource, agglutinative data-to-text.

In addition to evaluating models trained from scratch, we investigate finetuning pretrained models for T2X. IsiXhosa is not well represented in PLMs for text generation, so finetuning

does not guarantee good performance. Tokenisation-based massively multilingual PLMs fail to outperform SSPG. We only see gains from pretraining in byte-based PLMs adapted for African languages. We achieve further gains when we turn to the unconventional strategy of finetuning English → isiXhosa MT models on T2X, which achieve the best results overall. This surprising finding mirrors our results for dedicated architectures in that the well-established approach of finetuning PLMs is suboptimal for T2X.

To facilitate more nuanced evaluation with T2X we develop an extractive, data-oriented evaluation framework to evaluate how accurately the content in T2X data triples are verbalised in isiXhosa text generations. Applying this framework to the models in our experiments reveals tradeoffs between model capabilities: subword segmental models copy entities more accurately, while tokenisation-based pointer generators verbalise relations more effectively. We also perform a qualitative error analysis and an ablation study to further analyse the roles of different modelling components of SSPG.

Finally, we repeat our experiments on Finnish data-to-text and show that our T2X findings generalise to another agglutinative language. On the Finnish Hockey data-to-text dataset [68], SSPG outperforms baselines trained from scratch, improving the benchmark BLEU score of Kanerva et al. [68] by 1.2. As we for T2X, we also found that finetuning billingual English → Finnish MT models achieve the best performance overall. In summary, this chapter presented the following contributions:

1. We propose SSPG, which unifies subword segmentation, entity copying, and data-to-text generation in a single end-to-end neural architecture.

2. We propose unmixed decoding, a text generation algorithm for subword segmental models equipped with a copy mechanism.

3. We train SSPG on isiXhosa data-to-text and find that it outperforms tokenisation-based data-to-text architectures trained from scratch, as well as finetuned PLMs. SSPG is only outperformed by finetuned bilingual English → isiXhosa MT models.

4. We develop and apply an extractive evaluation framework for T2X, which reveals that SSPG copies data entities more accurately. A qualitative error analysis highlights the common morphological errors made by tokenisation-based models.

5. We repeat our experiments on Finnish data-to-text, replicating our findings on T2X to show that the success of SSPG generalises to another agglutinative language.

SSPG outperforms tokenisation-based dedicated data-to-text architectures on two agglutinative languages – isiXhosa and Finnish. It is a strong data-to-text model for low-resource agglutinative languages without existing high-quality pretrained models. In the face of such resource scarcity we also explored finetuning bilingual NMT models, which produced the strongest results overall and should be further investigated as an alternative to PLMs. However, PLMs and NMT models do not exist for many low-resource languages, in which case SSPG proves to be the strongest alternative for agglutinative languages. Our findings underscore the distinct challenge presented by low-resource, agglutinative data-to-text – neither well-established data-to-text architectures nor customary pretrained methodologies prove optimal.

7

Conclusion

In this thesis we have proposed three new neural models for text generation, all of which are specific instantiations of the more fundamental contribution of this thesis – the idea of subword segmental modelling. Our models are designed for three different tasks, namely language modelling, machine translation (MT), and data-to-text, and therefore extend different paradigms for neural sequence modelling, respectively decoder-only, encoder-decoder, and the pointer generator. The common thread running through this thesis is that all the models we have proposed share the core innovations which, considered together, we define as subword segmental modelling.

Firstly, their decoders are augmented with subnetworks that can model the probability of any subword sequence, including common subwords (through a frequency-based lexicon) and arbitrary character sequences (through a character-level decoder). Secondly, they are trained with a dynamic programming algorithm that computes the probability of a text sequence by marginalising over all possible subword segmentations of that sequence. The combination of this subword-driven architecture and training algorithm enables a new learning mechanism to emerge during training. In a forward pass, the model computes probabilities for all the possible permutations of subword units that a training instance can be segmented into. In a backward pass, the model updates its parameters to optimise the (marginalised) log-likelihood of the text sequence. As training unfolds, the model improves its predictions of future subwords based on preceding context, just as in standard tokenisation-based modelling.

For example, if trained on the target text “I was sleeping”, it would learn to assign a higher probability to the subword “ing” than “ed”, given the preceding context of “I was sleep”. The power of subword segmental modelling arises from the fact that the this subword-level learning would, in turn, also enable the model to adjust how it assigns probabilities to different subword segmentations. For example, it would learn to prefer “I was sleep–ing” over “I was sle–eping” if the affix “ing” was used in varied contexts in its training data. This is how a subword segmental model is able to learn subword segmentations that optimise its training objective.

In the remainder of this final chapter, we summarise the findings of preceding chapters, propose future directions for extending our work, and discuss the role of this thesis in the context of the wider NLP research landscape.

7.1 Summary of Findings

We have developed three variants of subword segmental models and applied them to three tasks across seven languages, with datasets spanning a range of training sizes. In this section we summarise our main findings for each model and task.

Chapter 3: Subword Segmental Language Modelling (SSLM)

- On small-scale, low-resource language modelling (training corpora of 500k to 3.4m tokens) SSLM obtains better test set perplexity-based scores than tokenisation-based language models, on average across the four Nguni languages.
- The subwords learned by SSLM are closer to morphemes than BPE and ULM subwords, but SSLM does sometimes oversegment morphemes and revert to character-level modelling.
- Among the new hyperparameters introduced by SSLM, the lexicon size and maximum segment length can greatly affect language modelling performance and should be tuned during development.

- A grid search on different hyperparameter settings shows that morphological segmentation F1 and perplexity are somewhat correlated i.e. subwords closer to morphemes lead to better language modelling.

Chapters 4 & 5: Subword Segmental Machine Translation (SSMT)

- SSMT mostly outperforms tokenisation-based MT models on agglutinative, conjunctive languages (including three Nguni languages), but falls short for disjunctive languages. SSMT obtains especially large gains (+2.5 chrF) on the extremely low-resource case (165k parallel sentences) of English → Siswati translation.

- The subwords learned by SSMT are closer to morphemes than BPE, ULM, and DPE subwords. SSMT also generalises better to an English → isiZulu test set constructed to evaluate morphological compositional generalisation (handling new combinations of known morphemes).

- SSMT does not improve cross-lingual transfer in multilingual models, but does prove beneficial when adapting an MT model from an unrelated, morphologically simple language (English → Afrikaans) to Nguni languages (English → isiXhosa/Siswati), via finetuning.

Chapter 6: Subword Segmental Pointer Generator (SSPG)

- SSPG outperforms dedicated data-to-text architectures, as well as finetuned text-to-text pretrained language models, on data-to-text for isiXhosa and Finnish (both agglutinative, conjunctive languages). It is only outperformed by finetuned bilingual MT models.

- SSPG combines isiXhosa prefix concatenation and entity copying more successfully than established data-to-text methods, but at the cost of performing worse than a standard pointer generator in verbalising isiXhosa subject-object relations.

7.2 Overarching Findings

This dissertation serves as a comprehensive investigation into the techniques of subword segmental modelling and its application to Nguni-language text generation. At this stage it is worth revisiting the hypotheses set out in Chapter 1.2, where we stated our thesis as "learning subword segmentation during training, instead of fixing it after preprocessing as in subword tokenisation, will prove beneficial for modelling low-resource, morphologically complex languages".

Our results, considered as a whole, establish subword segmental modelling as an effective alternative to subword tokenisation for developing text generation models for the Nguni languages. It fares better in the face of the data scarcity and morphological complexity that make the Nguni languages challenging from an NLP perspective. Our analyses also provide a solid understanding of the practical benefits, tradeoffs, and challenges of developing subword segmental models. In this section we summarise the general findings about subword segmental modelling, both positive and negative, that emerge across our study.

Advantages of Subword Segmental Modelling

- Subword segmental modelling consistently matches or outperforms the best performing tokenisation-based model on text generation tasks for the Nguni languages. This overcomes known issues around the inconsistency and unreliability of subword tokenisers on morphologically complex languages [174, 75].

- The largest performance gains obtained from subword segmental modelling are generally in settings where data is most limited, such as extremely low-resource languages (e.g. Siswati) and tasks with small datasets (e.g. isiXhosa data-to-text). We view the confluence of morphological complexity and data scarcity as the optimal application area for subword segmental modelling.

- During training, subword segmental modelling acquires substantial knowledge about the morphological systems of Nguni languages, despite no explicit supervision or access to external linguistic annotations. Subword segmental models can "discover"

some morphemes as subword units, compose them into words compositionally, and apply certain morphological rules (e.g. appending noun class prefixes) more effectively than standard tokenisation-based models.

Disadvantages of Subword Segmental Modelling

- Subword segmental language models take longer to train than tokenisation-based models. While the exact increase depends on hyperparameters (primarily the lexicon size and maximum segment length), in practice we found subword segmental modelling led to an increased training time of an order of magnitude ($\times 10$).

- Dynamic decoding, the text generation algorithm we proposed for subword segmental models in Chapter 4, is computationally inefficient compared to token-based beam search. In practice, it takes on average 300 times longer to generate a translation with SSMT than tokenisation-based MT (15 seconds instead of 0.05 seconds). Our unmixed decoding algorithm provides an efficient alternative, since it only increases decoding time in proportion to the number of mixture components ($\times 2$ for SSLM and SSMT, $\times 3$ for SSPG). However, unmixed decoding is only effective for certain tasks (e.g. data-to-text) and results in performance degradation for other tasks (e.g. MT), so dynamic decoding cannot always be avoided.

7.3 Future Work

Our research has outlined a framework for developing and applying subword segmental models, which can be extended and improved in future research.

7.3.1 Extension to More Architectures

We have presented three instantiations of subword segmental models, with architectures for language modelling, MT, and data-to-text. The techniques underlying subword segmental modelling are architecture-invariant – any neural sequence model can be adapted for subword

segmental modelling. We now highlight three opportunities for extending subword segmental modelling to architectures and training regimes unexplored in this thesis.

- The encoder-decoder of T5 [123] is trained on span denoising (reconstructing corrupted spans of the input text). This has proven effective as a pretraining strategy for a wide range of *text-to-text* tasks, including text summarisation, question answering, and text classification. A subword segmental variant of T5 would modify the decoder to reconstruct corrupted text spans by marginalising over all possible subword segmentations of a span. This would extend the encoder-decoder architecture of SSMT to more general text-to-text generation than MT. Low-resource languages are often limited to structured generation datasets (e.g. data-to-text, text summarisation), which is exactly where text-to-text models excel. Pretraining a subword segmental text-to-text model is a promising approach to text generation for the Nguni languages.

- Retrieval augmented generation (RAG) [85] has emerged as a popular method for incorporating external knowledge sources into text generation. This is achieved by augmenting the input to a neural text generation model with data retrieved from a knowledge base. The retrieval and generation is trained end-to-end, usually by jointly finetuning a PLM (the *generator*) and an embedding model (the *retriever*) that encodes chunks of the knowledge base for retrieval based on relevance to the query. Retrieval augmented subword segmental generation would entail jointly training a subword segmental decoder (as the generator) alongside a standard retriever, and adding the retrieved data to the input context of the original query. Such a model would be suitable for incorporating external knowledge into Nguni text generation.

- Extending subword segmental modelling to masked language modelling (MLM) [29] would require a variant of the MLM objective where the model reconstructs masked text by marginalising over all possible subword segmentations. This approach would produce contextual representations that encode information about different possible subword segmentations, and learn which segmentations optimise MLM prediction. MLM remains actively researched for the Nguni languages, since gains are still possible

on tasks liks POS tagging and text classification. A masked SSLM could improve performance on tasks that require morphological modelling, such as Nguni-language POS tagging where noun classes are indicated by prefixes. It could also enable more efficient MLM pretraining, given the performance gains of subword segmental modelling in low-resource settings. Downey et al. [32] propose a masked segmental language model, which is very similar to the masked SSLM we have described, except for minor differences in architecture (they do not include a subword lexicon) and training algorithm (they do not distinguish word boundaries, so their learned segments are not strictly subwords).

7.3.2 Application to More Languages and Tasks

The practical aim of this thesis has been to improve text generation models for the Nguni languages. As such, we have trained and evaluated models on tasks for which Nguni-language datasets exist. Given the consistent performance gains of subword segmental modelling across the four Nguni languages, as well as Finnish, our results suggest that these gains could generalise to other low-resource languages with similarly complex morphologies. We hope our work will encourage future researchers to test this claim by applying subword segmental modelling to other other languages. This would also open the door to applying the models proposed in this thesis to more datasets and tasks, beyond those available to the Nguni languages.

- The encoder-decoder architecture of SSMT is suitable for structured sequence-to-sequence text generation tasks other than MT, such as abstractive text summarisation, dialogue generation, and question answering. SSPG is suitable for text-to-text tasks in which parts of the input appear directly in the output text, such as abstractive text summarisation, headline generation, and more complex data-to-text tasks than the datasets used in this thesis (e.g. table-to-text or graph-to-text). These are tasks on which pointer generators have proven effective and the subword-level copying of SSPG could yield performance gains for morphologically rich languages.

- Recent breakthroughs in neural text generation for high-resource languages have enabled more open-ended generation than the tasks considered in this thesis [55]. This has been achieved through decoder-only LLMs than undergo large-scale pretraining [105, 150, 63], followed by instruction tuning [164, 108]. SSLM could be utilised in the same way, by pretraining a foundation SSLM and finetuning it for instruction following. This could enable more sample-efficient pretraining of LLMs for morphologically complex languages. Such an effort would require extensive computational resources, pretraining corpora larger than those currently available for the Nguni languages, and the development of training sets for Nguni-language instruction tuning.

- As an alternative to pretraining SSLMs from scratch, one could develop techniques that adapt tokenisation-based PLMs for subword segmental modelling. This would require transforming PLMs into subword segmental models, by utilising pretrained weights where possible and attaching newly initialised weights where necessary. There are many options for transforming PLMs to SSLMs, and doing so would require experimenting with different setups. One could partially initialise the weights of a Transformer-based SSLM with PLM weights (e.g. use the chracter-level encoders and decoders of ByT5 [168]) and then continue training with our subword segmental training objective. This approach would enable PLMs to optimise subword segmentation during finetuning, instead of relying on their initial subword tokenisation. This could improve finetuning of multilingual PLMs for low-resource languages. Multilingual PLMs cover several languages in their limited vocabulary. As a result, they often underrepresent and over-segment low-resource languages [161, 176, 112]. During adaptation, subword segmental modelling would adjust the language-agnostic tokenisation of multilingual PLMs to optimise performance for a specific target language.

7.3.3 Improving Computational Efficiency

Training A major disadvantage of subword segmental modelling in its current form is the high computational complexity of its training, which leads to approximately $10\times$ longer

training times than tokenization-based training in practice. The source of this complexity is the marginalisation over all possible subword segmentations which, despite our dynamic programming algorithm, introduces additional probability computations. While some increased training overhead is unavoidable for subword segmental modelling, it would be fruitful to explore ways of making the training algorithm more efficient than it currently is. This could be achieved by approximating the marginalisation during training, as done with sampling-based methods by Cao and Rimell [15] and Chirkova et al. [19]. Alternatively, if a way was found to induce sparsity on the distribution of subword segmentations, this could effectively discard subspaces of possible subword segmentations during training (assigning them zero probability), which could in turn lead to fewer probability computations.

Decoding The limitation that, perhaps more than any other, would prohibit the adoption of subword segmental models for some Nguni-language tasks is the approximately 300-fold time increase introduced by dynamic decoding for text generation. While the more efficient unmixed decoding leads to better performane for some tasks (e.g. data-to-text), for other tasks (e.g. MT) dynamic decoding is required to obtain the performance gains that make subword segmental modelling attractive in the first place. Subword segmental modelling would be greatly improved as a modelling framework with the introduction of decoding algorithms that improve efficiency without losing performance. This could be achieved by strategies similar to those we put forth as potential solutions for efficient training, namely sophisticated sampling strategies and utilising sparsity of the subword segmentation distribution during decoding.

7.3.4 Incorporating Linguistic Annotation

We have proposed subword segmental modelling as an end-to-end system trained to optimise task performance. This has the advantage of requiring no linguistically annotated data, which is often lacking for low-resource languages. Trained to generate the raw text data for a specific task, subword segmental modelling is not limited by morphological boundaries and can discover the subword units that optimise performance. However, we showed that

SSLM and SSMT end up learning subword units that partially align with morphemes. Since morphological knowledge emerges implicitly in this way and might contribute to the performance gains of subword segmental modelling, it is worth exploring ways of incorporating morphologically annotations into subword segmental training. Previous work has shown that incorporating morphological analyses into neural modelling [103] and morpheme boundaries into subword tokenisation [60, 59, 10] can improve model performance. Given the availability of such morphologically annotated data for the Nguni languages, this could similarly benefit subword segmental modelling. For example, subword segment boundaries that cross morpheme boundaries could be prohibited, or morphological segmentation accuracy could be jointly optimised alongside text generation (effectively biasing the model towards morphemes as subwords). This would not give up the end-to-end performance optimisation of subword segmental modelling, but instead augment it with potentially useful inductive biases.

7.4 Closing Discussion

In conclusion, we would like to take a step back and consider our contributions within the broader context of NLP research. The work presented in this thesis was conducted during a time of great progress and upheaval in NLP. ChatGPT was released around the midway point of this project. It signified the culmination of a continuous line of research tracing back to the success of sequence-to-sequence models, only ten years ago. The intermediate decade has been characterised by certain trends which have taken the field by storm and advanced NLP research beyond expectations. To take a bird's eye perspective on this thesis, we now consider two of these trends and highlight how our findings go along with the first and push back against the second.

End-to-end learning Firstly, since the advent of sequence-to-sequence learning a decade ago, there has been a trend towards NLP systems that are trained end-to-end. Instead of explicitly separating modelling into distinct subsystems, as was done in the past, the modern approach is to learn the entire task via supervised training. The traditional NLP pipeline,

which incorporated linguistic resources and symbolic approaches, has been replaced by an end-to-end approach for most tasks. The only step which has not been absorbed into end-to-end learning is subword segmentation. In modern NLP, subword segmentation is universally applied as a distinct preprocessing step separated from model training. In this sense, our work follows the trend towards end-to-end learning by extending it to subword segmentation, one of the few aspects of modelling that is not yet usually learned in an end-to-end fashion. We have shown the benefits of including subword segmentation under the umbrella of end-to-end training. For the Nguni languages, subword segmentation is important enough to leverage the full potential of neural network training and optimisation.

Language-agnostic modelling Secondly, NLP has moved away from developing language-specific models to applying the same modelling techniques to all languages. This is related to the trend of end-to-end learning, the idea being that a sufficiently powerful sequence model (currently the Transformer and previously the LSTM) could generalise across languages. Multilingual modelling takes this idea even further, instantiating a single model for multiple languages. Our research pushes back against language-agnostic modelling. We have shown that developing language-specific models, designed to more effectively model particular language characteristics, can improve performance over language-invariant models. The standard approach of casting subword segmentation as a preprocessing might be sufficient for some languages, but it is suboptimal for the Nguni languages. Subword segmental modelling takes the distinct complex morphology of Nguni languages into account and shows that doing so can benefit text generation. We have not reached the point where a single neural network architecture is optimal across all languages and tasks. The linguistic diversity of the world still presents an opportunity for innovation in NLP modelling.

Bibliography

[1] Adelani, D., Alabi, J., Fan, A., Kreutzer, J., Shen, X., Reid, M., Ruiter, D., Klakow, D., Nabende, P., Chang, E., Gwadabe, T., Sackey, F., Dossou, B. F. P., Emezue, C., Leong, C., Beukman, M., Muhammad, S., Jarso, G., Yousuf, O., Niyongabo Rubungo, A., Hacheme, G., Wairagala, E. P., Nasir, M. U., Ajibade, B., Ajayi, T., Gitau, Y., Abbott, J., Ahmed, M., Ochieng, M., Aremu, A., Ogayo, P., Mukiibi, J., Ouoba Kabore, F., Kalipe, G., Mbaye, D., Tapo, A. A., Memdjokam Koagne, V., Munkoh-Buabeng, E., Wagner, V., Abdulmumin, I., Awokoya, A., Buzaaba, H., Sibanda, B., Bukula, A., and Manthalu, S. (2022a). A few thousand translations go a long way! leveraging pre-trained models for African news translation. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V., editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3053–3070, Seattle, United States. Association for Computational Linguistics.

[2] Adelani, D., Alam, M. M. I., Anastasopoulos, A., Bhagia, A., Costa-jussà, M. R., Dodge, J., Faisal, F., Federmann, C., Fedorova, N., Guzmán, F., Koshelev, S., Maillard, J., Marivate, V., Mbuya, J., Mourachko, A., Saleem, S., Schwenk, H., and Wenzek, G. (2022b). Findings of the WMT'22 shared task on large-scale machine translation evaluation for African languages. In Koehn, P., Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Freitag, M., Graham, Y., Grundkiewicz, R., Guzman, P., Haddow, B., Huck, M., Jimeno Yepes, A., Kocmi, T., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., Negri, M., Névéol, A., Neves, M., Popel, M., Turchi, M., and Zampieri, M., editors, *Proceedings*

of the Seventh Conference on Machine Translation (WMT), pages 773–800, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

[3] Adelani, D. I., Ojo, J., Azime, I. A., Zhuang, J. Y., Alabi, J. O., He, X., Ochieng, M., Hooker, S., Bukula, A., Lee, E.-S. A., Chukwuneke, C., Buzaaba, H., Sibanda, B., Kalipe, G., Mukiibi, J., Kabongo, S., Yuehgoh, F., Setaka, M., Ndolela, L., Odu, N., Mabuya, R., Muhammad, S. H., Osei, S., Samb, S., Guge, T. K., and Stenetorp, P. (2024). Irokobench: A new benchmark for african languages in the age of large language models. *arXiv:2406.03368*.

[4] Aharoni, R., Johnson, M., and Firat, O. (2019). Massively multilingual neural machine translation. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

[5] Akyurek, E. and Andreas, J. (2021). Lexicon learning for few shot sequence modeling. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4934–4946, Online. Association for Computational Linguistics.

[6] Alabi, J. O., Adelani, D. I., Mosbach, M., and Klakow, D. (2022). Adapting pre-trained language models to African languages via multilingual adaptive fine-tuning. In Calzolari, N., Huang, C.-R., Kim, H., Pustejovsky, J., Wanner, L., Choi, K.-S., Ryu, P.-M., Chen, H.-H., Donatelli, L., Ji, H., Kurohashi, S., Paggio, P., Xue, N., Kim, S., Hahm, Y., He, Z., Lee, T. K., Santus, E., Bond, F., and Na, S.-H., editors, *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4336–4349, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

[7] Araabi, A. and Monz, C. (2020). Optimizing transformer for low-resource neural machine translation. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th*

International Conference on Computational Linguistics, pages 3429–3435, Barcelona, Spain (Online). International Committee on Computational Linguistics.

[8] Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[9] Bapna, A., Caswell, I., Kreutzer, J., Firat, O., van Esch, D., Siddhant, A., Niu, M., Baljekar, P. N., Garcia, X., Macherey, W., Breiner, T., Axelrod, V. S., Riesa, J., Cao, Y., Chen, M., Macherey, K., Krikun, M., Wang, P., Gutkin, A., Shah, A., Huang, Y., Chen, Z., Wu, Y., and Hughes, M. R. (2022). Building machine translation systems for the next thousand languages. Technical report, Google Research.

[10] Bauwens, T. and Delobelle, P. (2024). BPE-knockout: Pruning pre-existing BPE tokenisers with backwards-compatible morphological semi-supervision. In Duh, K., Gomez, H., and Bethard, S., editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5810–5832, Mexico City, Mexico. Association for Computational Linguistics.

[11] Bender, E. (2019). The #benderrule: On naming the languages we study and why it matters. *The Gradient*. [Online; accessed 13-September-2021].

[12] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

[13] Buys, J. and Blunsom, P. (2018). Neural syntactic generative models with exact marginalization. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 942–952, New Orleans, Louisiana. Association for Computational Linguistics.

[14] Cai, D., Wang, Y., Li, H., Lam, W., and Liu, L. (2021). Neural machine translation with monolingual translation memory. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7307–7318, Online. Association for Computational Linguistics.

[15] Cao, K. and Rimell, L. (2021). You should evaluate your language model on marginal likelihood over tokenisations. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2104–2114, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

[16] Carrión, S. and Casacuberta, F. (2022). On the effectiveness of quasi character-level models for machine translation. In Duh, K. and Guzmán, F., editors, *Proceedings of the 15th biennial conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 131–143, Orlando, USA. Association for Machine Translation in the Americas.

[17] Castro Ferreira, T., Gardent, C., Ilinykh, N., van der Lee, C., Mille, S., Moussallem, D., and Shimorina, A. (2020). The 2020 bilingual, bi-directional WebNLG+ shared task: Overview and evaluation results (WebNLG+ 2020). In Castro Ferreira, T., Gardent, C., Ilinykh, N., van der Lee, C., Mille, S., Moussallem, D., and Shimorina, A., editors, *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 55–76, Dublin, Ireland (Virtual). Association for Computational Linguistics.

[18] Castro Ferreira, T., van der Lee, C., van Miltenburg, E., and Krahmer, E. (2019). Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 552–562, Hong Kong, China. Association for Computational Linguistics.

[19] Chirkova, N., Kruszewski, G., Rozen, J., and Dymetman, M. (2023). Should you marginalize over possible tokenizations? In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–12, Toronto, Canada. Association for Computational Linguistics.

[20] Chung, H. W., Garrette, D., Tan, K. C., and Riesa, J. (2020). Improving multilingual models with language-clustered vocabularies. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4536–4546, Online. Association for Computational Linguistics.

[21] Chung, J., Kannappan, P., Ng, C., and Sahoo, P. (1989). Measures of distance between probability distributions. *Journal of Mathematical Analysis and Applications*, 138(1):280–292.

[22] Clark, J. H., Garrette, D., Turc, I., and Wieting, J. (2022). Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

[23] Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

[24] Cotterell, R., Vieira, T., and Schütze, H. (2016). A joint model of orthography and morphological segmentation. In Knight, K., Nenkova, A., and Rambow, O., editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 664–669, San Diego, California. Association for Computational Linguistics.

[25] Creutz, M. and Lagus, K. (2007). Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(1).

[26] Cripwell, L., Belz, A., Gardent, C., Gatt, A., Borg, C., Borg, M., Judge, J., Lorandi, M., Nikiforovskaya, A., and Soto Martinez, W. (2023). The 2023 WebNLG shared task on low resource languages. overview and evaluation results (WebNLG 2023). In Gatt, A., Gardent, C., Cripwell, L., Belz, A., Borg, C., Erdem, A., and Erdem, E., editors, *Proceedings of the Workshop on Multimodal, Multilingual Natural Language Generation and Multilingual WebNLG Challenge (MM-NLG 2023)*, pages 55–66, Prague, Czech Republic. Association for Computational Linguistics.

[27] Dabre, R., Chu, C., and Kunchukuttan, A. (2020). A survey of multilingual neural machine translation. *Association for Computing Machinery (ACM) Computing Surveys (CSUR)*, 53(5).

[28] Dabre, R., Shrotriya, H., Kunchukuttan, A., Puduppully, R., Khapra, M., and Kumar, P. (2022). IndicBART: A pre-trained model for indic natural language generation. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1849–1863, Dublin, Ireland. Association for Computational Linguistics.

[29] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume*

1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

[30] Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, page 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[31] Dossou, B. F. P., Tonja, A. L., Yousuf, O., Osei, S., Oppong, A., Shode, I., Awoyomi, O. O., and Emezue, C. (2022). AfroLM: A self-active learning-based multilingual pretrained language model for 23 African languages. In Fan, A., Gurevych, I., Hou, Y., Kozareva, Z., Luccioni, S., Sadat Moosavi, N., Ravi, S., Kim, G., Schwartz, R., and Rücklé, A., editors, *Proceedings of The Third Workshop on Simple and Efficient Natural Language Processing (SustaiNLP)*, pages 52–64, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

[32] Downey, C., Xia, F., Levow, G.-A., and Steinert-Threlkeld, S. (2022). A masked segmental language model for unsupervised natural language segmentation. In Nicolai, G. and Chodroff, E., editors, *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 39–50, Seattle, Washington. Association for Computational Linguistics.

[33] Eberhard, D. M., Simons, G. F., , and Fenning, C. D. (2019). *Ethnologue: Languages of the World*. SIL International, 22nd edition.

[34] Ebrahimi, A., Mager, M., Oncevay, A., Chaudhary, V., Chiruzzo, L., Fan, A., Ortega, J., Ramos, R., Rios, A., Meza Ruiz, I. V., Giménez-Lugo, G., Mager, E., Neubig, G., Palmer, A., Coto-Solano, R., Vu, T., and Kann, K. (2022). AmericasNLI: Evaluating zero-shot natural language understanding of pretrained multilingual models in truly low-resource languages. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6279–6299, Dublin, Ireland. Association for Computational Linguistics.

[35] Edman, L., Sarti, G., Toral, A., Noord, G. v., and Bisazza, A. (2024). Are character-level translations worth the wait? comparing ByT5 and mT5 for machine translation. *Transactions of the Association for Computational Linguistics*, 12:392–410.

[36] Edman, L., Toral, A., and van Noord, G. (2022a). Patching leaks in the charformer for efficient character-level generation. *arXiv:2205.14086*.

[37] Edman, L., Toral, A., and van Noord, G. (2022b). Subword-delimited downsampling for better character-level translation. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 981–992, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

[38] Eiselen, R. and Puttkammer, M. (2014). Developing text resources for ten South African languages. In Calzolari, N., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3698–3703, Reykjavik, Iceland. European Language Resources Association (ELRA).

[39] Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, 14(2):179–211.

[40] Eskander, R., Klavans, J., and Muresan, S. (2019). Unsupervised morphological segmentation for low-resource polysynthetic languages. In Nicolai, G. and Cotterell, R., editors, *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 189–195, Florence, Italy. Association for Computational Linguistics.

[41] Feng, Y., Zhang, S., Zhang, A., Wang, D., and Abel, A. (2017). Memory-augmented neural machine translation. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1390–1399, Copenhagen, Denmark. Association for Computational Linguistics.

[42] Firat, O., Cho, K., and Bengio, Y. (2016). Multi-way, multilingual neural machine translation with a shared attention mechanism. In Knight, K., Nenkova, A., and Rambow,

O., editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 866–875, San Diego, California. Association for Computational Linguistics.

[43] Fodor, J. A. and Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1):3–71.

[44] Gage, P. (1994). A new algorithm for data compression. *The C Users Journal*, 12(2):23–38.

[45] Gardent, C., Shimorina, A., Narayan, S., and Perez-Beltrachini, L. (2017). Creating training corpora for NLG micro-planners. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 179–188, Vancouver, Canada. Association for Computational Linguistics.

[46] Gatt, A. and Krahmer, E. (2017). Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

[47] Gaustad, T. and Puttkammer, M. J. (2022). Linguistically annotated dataset for four official South African languages with a conjunctive orthography: IsiNdebele, isiXhosa, isiZulu, and Siswati. *Data in Brief*, 41:107994.

[48] Goldberg, Y. (2016). A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57(1):345–420.

[49] Goyal, N., Gao, C., Chaudhary, V., Chen, P.-J., Wenzek, G., Ju, D., Krishnan, S., Ranzato, M., Guzmán, F., and Fan, A. (2022). The Flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

[50] Graves, A. (2014). Generating sequences with recurrent neural networks. *arXiv:1308.0850*.

[51] Graves, A. (2017). Adaptive computation time for recurrent neural networks. *arXiv:1603.08983*.

[52] Gulcehre, C., Ahn, S., Nallapati, R., Zhou, B., and Bengio, Y. (2016). Pointing the unknown words. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 140–149, Berlin, Germany. Association for Computational Linguistics.

[53] Ha, T.-L., Niehues, J., and Waibel, A. (2016). Toward multilingual neural machine translation with universal encoder and decoder. In Cettolo, M., Niehues, J., Stüker, S., Bentivogli, L., Cattoni, R., and Federico, M., editors, *Proceedings of the 13th International Conference on Spoken Language Translation*, Seattle, Washington D.C. International Workshop on Spoken Language Translation.

[54] Haddow, B., Bawden, R., Miceli Barone, A. V., Helcl, J., and Birch, A. (2022). Survey of low-resource machine translation. *Computational Linguistics*, 48(3):673–732.

[55] Hadi, M. U., Al Tashi, Q., Shah, A., Qureshi, R., Muneer, A., Irfan, M., Zafar, A., Shaikh, M. B., Akhtar, N., Wu, J., et al. (2024). Large language models: a comprehensive survey of its applications, challenges, limitations, and future prospects. *Authorea Preprints*.

[56] He, X., Haffari, G., and Norouzi, M. (2020). Dynamic programming encoding for subword segmentation in neural machine translation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3042–3051, Online. Association for Computational Linguistics.

[57] Hedderich, M. A., Lange, L., Adel, H., Strötgen, J., and Klakow, D. (2021). A survey on recent approaches for natural language processing in low-resource scenarios. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2545–2568, Online. Association for Computational Linguistics.

[58] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

[59] Hofmann, V., Pierrehumbert, J., and Schütze, H. (2021). Superbizarre is not superb: Derivational morphology improves BERT's interpretation of complex words. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608, Online. Association for Computational Linguistics.

[60] Huck, M., Riess, S., and Fraser, A. (2017). Target-side word segmentation strategies for neural machine translation. In Bojar, O., Buck, C., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., and Kreutzer, J., editors, *Proceedings of the Second Conference on Machine Translation*, pages 56–67, Copenhagen, Denmark. Association for Computational Linguistics.

[61] Hupkes, D., Dankers, V., Mul, M., and Bruni, E. (2020). Compositionality decomposed: How do neural networks generalise? (extended abstract). In Bessiere, C., editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 5065–5069. International Joint Conferences on Artificial Intelligence Organization. Journal track.

[62] Hupkes, D., Giulianelli, M., Dankers, V., Artetxe, M., Elazar, Y., Pimentel, T., Christodoulopoulos, C., Lasri, K., Saphra, N., Sinclair, A., Ulmer, D., Schottmann, F., Batsuren, K., Sun, K., Sinha, K., Khalatbari, L., Ryskina, M., Frieske, R., Cotterell, R., and Jin, Z. (2022). State-of-the-art generalisation research in NLP: a taxonomy and review. *CoRR*.

[63] Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A.,

Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2023). Mistral 7b. *arXiv:2310.06825*.

[64] Jones, C. R. and Bergen, B. K. (2024). People cannot distinguish GPT-4 from a human in a Turing test. *arXiv:2405.08007*.

[65] Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N., and Wu, Y. (2016). Exploring the limits of language modeling. *arXiv:1602.02410*.

[66] Jude Ogundepo, O., Oladipo, A., Adeyemi, M., Ogueji, K., and Lin, J. (2022). AfriTeVA: Extending ?small data? pretraining approaches to sequence-to-sequence models. In Cherry, C., Fan, A., Foster, G., Haffari, G. R., Khadivi, S., Peng, N. V., Ren, X., Shareghi, E., and Swayamdipta, S., editors, *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 126–135, Hybrid. Association for Computational Linguistics.

[67] Kale, M. and Rastogi, A. (2020). Text-to-text pre-training for data-to-text tasks. In Davis, B., Graham, Y., Kelleher, J., and Sripada, Y., editors, *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

[68] Kanerva, J., Rönnqvist, S., Kekki, R., Salakoski, T., and Ginter, F. (2019). Template-free data-to-text generation of Finnish sports news. In Hartmann, M. and Plank, B., editors, *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 242–252, Turku, Finland. Linköping University Electronic Press.

[69] Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020). Scaling laws for neural language models. *arXiv:2001.08361*.

[70] Kawakami, K., Dyer, C., and Blunsom, P. (2019). Learning to discover, ground and use words with segmental neural language models. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6429–6441, Florence, Italy. Association for Computational Linguistics.

[71] Keysers, D., Schärli, N., Scales, N., Buisman, H., Furrer, D., Kashubin, S., Momchev, N., Sinopalnikov, D., Stafiniak, L., Tihon, T., Tsarkov, D., Wang, X., van Zee, M., and Bousquet, O. (2020). Measuring compositional generalization: A comprehensive method on realistic data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020, Conference Track Proceedings*.

[72] Kim, N. and Linzen, T. (2020). COGS: A compositional generalization challenge based on semantic interpretation. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9087–9105, Online. Association for Computational Linguistics.

[73] Kim, Y., Jernite, Y., Sontag, D., and Rush, A. M. (2016). Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, page 2741–2749. AAAI Press.

[74] Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[75] Klein, S. and Tsarfaty, R. (2020). Getting the ##life out of living: How adequate are word-pieces for modelling complex morphology? In Nicolai, G., Gorman, K., and Cotterell, R., editors, *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 204–209, Online. Association for Computational Linguistics.

[76] Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In Lin, D. and Wu, D., editors, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

[77] Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In Ananiadou,

S., editor, *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

[78] Kong, L., Dyer, C., and Smith, N. A. (2016). Segmental recurrent neural networks. In Bengio, Y. and LeCun, Y., editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

[79] Kreutzer, J., Caswell, I., Wang, L., Wahab, A., van Esch, D., Ulzii-Orshikh, N., Tapo, A. A., Subramani, N., Sokolov, A., Sikasote, C., Setyawan, M., Sarin, S., Samb, S., Sagot, B., Rivera, C. E., Rios, A., Papadimitriou, I., Osei, S., Suárez, P. J. O., Onome'Orife, I. F., Ogueji, K., Niyongabo, R. A., Nguyen, T., Müller, M., Müller, A., Muhammad, S. H., Muhammad, N., Mnyakeni, A., Mirzakhalov, J., Matangira, T., Leong, C., Lawson, N., Kudugunta, S., Jernite, Y., Jenny, M., Firat, O., Dossou, B. F. P., Dlamini, S., de Silva, N., Çabuk Ballı, S., Biderman, S., Battisti, A., Baruwa, A., Bapna, A., Baljekar, P., Azime, I. A., Awokoya, A., Ataman, D., Ahia, O., Ahia, O., Agrawal, S., and Adeyemi, M. (2021). Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics (TACL)*.

[80] Kreutzer, J. and Sokolov, A. (2018). Learning to segment inputs for NMT favors character-level processing. In Turchi, M., Niehues, J., and Frederico, M., editors, *Proceedings of the 15th International Conference on Spoken Language Translation*, pages 166–172, Brussels. International Conference on Spoken Language Translation.

[81] Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

[82] Kudo, T. and Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Blanco, E. and Lu, W., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

[83] Lake, B. and Baroni, M. (2018). Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In Dy, J. and Krause, A., editors, *35th International Conference on Machine Learning, ICML 2018*, 35th International Conference on Machine Learning, ICML 2018, pages 4487–4499. International Machine Learning Society (IMLS).

[84] Lavie, A. and Agarwal, A. (2007). METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In Callison-Burch, C., Koehn, P., Fordyce, C. S., and Monz, C., editors, *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

[85] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

[86] Libovický, J., Schmid, H., and Fraser, A. (2022). Why don't people use character-level machine translation? In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2470–2485, Dublin, Ireland. Association for Computational Linguistics.

[87] Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

[88] Ma, S., Yang, P., Liu, T., Li, P., Zhou, J., and Sun, X. (2019). Key fact as pivot: A two-stage model for low resource table-to-text generation. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2047–2057, Florence, Italy. Association for Computational Linguistics.

[89] Maronikolakis, A., Dufter, P., and Schütze, H. (2021). Wine is not v i n. on the compatibility of tokenizations across languages. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2382–2399, Punta Cana, Dominican Republic. Association for Computational Linguistics.

[90] McKeown, K. R. (1985). *Text generation: using discourse strategies and focus constraints to generate natural language text*. Cambridge University Press, USA.

[91] Mesham, S., Hayward, L., Shapiro, J., and Buys., J. (2021). Low-resource language modelling of south african languages. In *Proceedings of the Second Southern African Conference for Artificial Intelligence Research (SACAIR)*, Online. Springer.

[92] Meyer, F. and Buys, J. (2024). Triples-to-isiXhosa (T2X): Addressing the challenges of low-resource agglutinative data-to-text generation. In Calzolari, N., Kan, M.-Y., Hoste, V., Lenci, A., Sakti, S., and Xue, N., editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 16841–16854, Torino, Italia. ELRA and ICCL.

[93] Meyer, F., Song, H., Chakrabarty, A., Buys, J., Dabre, R., and Tanaka, H. (2024). NGLUEni: Benchmarking and adapting pretrained language models for nguni languages. In Calzolari, N., Kan, M.-Y., Hoste, V., Lenci, A., Sakti, S., and Xue, N., editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 12247–12258, Torino, Italia. ELRA and ICCL.

[94] Mielke, S. J. (2016). Language diversity in ACL 2004 - 2016. [Online; accessed 13-September-2021].

[95] Mielke, S. J., Alyafeai, Z., Salesky, E., Raffel, C., Dey, M., Gallé, M., Raja, A., Si, C., Lee, W. Y., Sagot, B., and Tan, S. Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP. *arXiv:2112.10508*.

[96] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

[97] Mitchell, M. (2024). The turing test and our shifting conceptions of intelligence. *Science*, 385(6710):eadq9356.

[98] Moeng, T., Reay, S., Daniels, A., and Buys., J. (2021). Canonical and surface morphological segmentation for nguni languages. In *Proceedings of the Second Southern African Conference for Artificial Intelligence Research (SACAIR)*, pages 125–139, Online. Springer.

[99] Mzamo, L., Helberg, A., and Bosch, S. (2019a). Towards an unsupervised morphological segmenter for isixhosa. In *Southern African Universities Power Engineering Conference/Robotics and Mechatronics/Pattern Recognition Association of South Africa (SAUPEC/RobMech/PRASA)*, pages 166–170.

[100] Mzamo, L., Helberg, A., and Bosch, S. E. (2019b). Evaluation of combined bidirectional branching entropy language models for morphological segmentation of isixhosa. In *South African Forum of Artificial Intelligence Research*.

[101] Nan, L., Radev, D., Zhang, R., Rau, A., Sivaprasad, A., Hsieh, C., Tang, X., Vyas, A., Verma, N., Krishna, P., Liu, Y., Irwanto, N., Pan, J., Rahman, F., Zaidi, A., Mutuma, M., Tarabar, Y., Gupta, A., Yu, T., Tan, Y. C., Lin, X. V., Xiong, C., Socher, R., and Rajani, N. F. (2021). DART: Open-domain structured data record to text generation. In Toutanova,

K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447, Online. Association for Computational Linguistics.

[102] Novikova, J., Dušek, O., and Rieser, V. (2017). The E2E dataset: New challenges for end-to-end generation. In Jokinen, K., Stede, M., DeVault, D., and Louis, A., editors, *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany. Association for Computational Linguistics.

[103] Nzeyimana, A. and Niyongabo Rubungo, A. (2022). KinyaBERT: a morphology-aware Kinyarwanda language model. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5347–5363, Dublin, Ireland. Association for Computational Linguistics.

[104] Ojo, J., Ogueji, K., Stenetorp, P., and Adelani, D. I. (2024). How good are Large Language Models on African Languages? *arXiv:2311.07978*.

[105] OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., Bello, I., Berdine, J., Bernadett-Shapiro, G., Berner, C., Bogdonoff, L., Boiko, O., Boyd, M., Brakman, A.-L., Brockman, G., Brooks, T., Brundage, M., Button, K., Cai, T., Campbell, R., Cann, A., Carey, B., Carlson, C., Carmichael, R., Chan, B., Chang, C., Chantzis, F., Chen, D., Chen, S., Chen, R., Chen, J., Chen, M., Chess, B., Cho, C., Chu, C., Chung, H. W., Cummings, D., Currier, J., Dai, Y., Decareaux, C., Degry, T., Deutsch, N., Deville, D., Dhar, A., Dohan, D., Dowling, S., Dunning, S., Ecoffet, A., Eleti, A., Eloundou, T., Farhi, D., Fedus, L., Felix, N., Fishman, S. P., Forte, J., Fulford, I., Gao, L., Georges, E., Gibson, C., Goel, V., Gogineni, T., Goh, G., Gontijo-Lopes, R., Gordon, J., Grafstein, M., Gray, S., Greene, R., Gross, J., Gu, S. S., Guo, Y., Hallacy, C., Han, J., Harris, J., He, Y., Heaton, M., Heidecke, J., Hesse, C., Hickey, A., Hickey, W., Hoeschele, P., Houghton, B., Hsu, K.,

Hu, S., Hu, X., Huizinga, J., Jain, S., Jain, S., Jang, J., Jiang, A., Jiang, R., Jin, H., Jin, D., Jomoto, S., Jonn, B., Jun, H., Kaftan, T., Łukasz Kaiser, Kamali, A., Kanitscheider, I., Keskar, N. S., Khan, T., Kilpatrick, L., Kim, J. W., Kim, C., Kim, Y., Kirchner, J. H., Kiros, J., Knight, M., Kokotajlo, D., Łukasz Kondraciuk, Kondrich, A., Konstantinidis, A., Kosic, K., Krueger, G., Kuo, V., Lampe, M., Lan, I., Lee, T., Leike, J., Leung, J., Levy, D., Li, C. M., Lim, R., Lin, M., Lin, S., Litwin, M., Lopez, T., Lowe, R., Lue, P., Makanju, A., Malfacini, K., Manning, S., Markov, T., Markovski, Y., Martin, B., Mayer, K., Mayne, A., McGrew, B., McKinney, S. M., McLeavey, C., McMillan, P., McNeil, J., Medina, D., Mehta, A., Menick, J., Metz, L., Mishchenko, A., Mishkin, P., Monaco, V., Morikawa, E., Mossing, D., Mu, T., Murati, M., Murk, O., Mély, D., Nair, A., Nakano, R., Nayak, R., Neelakantan, A., Ngo, R., Noh, H., Ouyang, L., O'Keefe, C., Pachocki, J., Paino, A., Palermo, J., Pantuliano, A., Parascandolo, G., Parish, J., Parparita, E., Passos, A., Pavlov, M., Peng, A., Perelman, A., de Avila Belbute Peres, F., Petrov, M., de Oliveira Pinto, H. P., Michael, Pokorny, Pokrass, M., Pong, V. H., Powell, T., Power, A., Power, B., Proehl, E., Puri, R., Radford, A., Rae, J., Ramesh, A., Raymond, C., Real, F., Rimbach, K., Ross, C., Rotsted, B., Roussez, H., Ryder, N., Saltarelli, M., Sanders, T., Santurkar, S., Sastry, G., Schmidt, H., Schnurr, D., Schulman, J., Selsam, D., Sheppard, K., Sherbakov, T., Shieh, J., Shoker, S., Shyam, P., Sidor, S., Sigler, E., Simens, M., Sitkin, J., Slama, K., Sohl, I., Sokolowsky, B., Song, Y., Staudacher, N., Such, F. P., Summers, N., Sutskever, I., Tang, J., Tezak, N., Thompson, M. B., Tillet, P., Tootoonchian, A., Tseng, E., Tuggle, P., Turley, N., Tworek, J., Uribe, J. F. C., Vallone, A., Vijayvergiya, A., Voss, C., Wainwright, C., Wang, J. J., Wang, A., Wang, B., Ward, J., Wei, J., Weinmann, C., Welihinda, A., Welinder, P., Weng, J., Weng, L., Wiethoff, M., Willner, D., Winter, C., Wolrich, S., Wong, H., Workman, L., Wu, S., Wu, J., Wu, M., Xiao, K., Xu, T., Yoo, S., Yu, K., Yuan, Q., Zaremba, W., Zellers, R., Zhang, C., Zhang, M., Zhao, S., Zheng, T., Zhuang, J., Zhuk, W., and Zoph, B. (2024). GPT-4 technical report. *arXiv:2303.08774*.

[106] Ortega, J. E., Castro Mamani, R., and Cho, K. (2020). Neural machine translation with a polysynthetic low resource language. *Machine Translation*, 34(4):325–346.

[107] Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In Ammar, W., Louis, A., and Mostafazadeh, N., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

[108] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

[109] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In Isabelle, P., Charniak, E., and Lin, D., editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

[110] Park, H. H., Zhang, K. J., Haley, C., Steimel, K., Liu, H., and Schwartz, L. (2021). Morphology matters: A multilingual language modeling analysis. *Transactions of the Association for Computational Linguistics*, 9:261–276.

[111] Partee, B. (1984). Compositionality. *Varieties of formal semantics*, pages 281—-311.

[112] Patil, V., Talukdar, P., and Sarawagi, S. (2022). Overlap-based vocabulary generation improves cross-lingual transfer among related languages. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 219–233, Dublin, Ireland. Association for Computational Linguistics.

[113] Pires, T., Schlinger, E., and Garrette, D. (2019). How multilingual is multilingual BERT? In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th*

Annual Meeting of the Association for Computational Linguistics, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

[114] Poon, H., Cherry, C., and Toutanova, K. (2009). Unsupervised morphological segmentation with log-linear models. In Ostendorf, M., Collins, M., Narayanan, S., Oard, D. W., and Vanderwende, L., editors, *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 209–217, Boulder, Colorado. Association for Computational Linguistics.

[115] Popović, M. (2015). chrF: character n-gram F-score for automatic MT evaluation. In Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Hokamp, C., Huck, M., Logacheva, V., and Pecina, P., editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

[116] Popović, M. (2017). chrF++: words helping character n-grams. In Bojar, O., Buck, C., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., and Kreutzer, J., editors, *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

[117] Pretorius, R., Berg, A., Pretorius, L., and Viljoen, B. (2009). Setswana tokenisation and computational verb morphology: Facing the challenge of a disjunctive orthography. In Levin, L., Kiango, J., Klavans, J., De Pauw, G., de Schryver, G.-M., and Wagacha, P. W., editors, *Proceedings of the First Workshop on Language Technologies for African Languages*, pages 66–73, Athens, Greece. Association for Computational Linguistics.

[118] Provilkov, I., Emelianenko, D., and Voita, E. (2020). BPE-dropout: Simple and effective subword regularization. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

[119] Puduppully, R., Dong, L., and Lapata, M. (2019). Data-to-text generation with content selection and planning. In *Proceedings of the Thirty-Third AAAI Conference on*

Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

[120] Puduppully, R. and Lapata, M. (2021). Data-to-text Generation with Macro Planning. *Transactions of the Association for Computational Linguistics*, 9:510–527.

[121] Puduppully, R. S. (2021). *Data-to-text Generation with Neural Planning*. Doctor of philosophy, University of Edinburgh, Edinburgh, UK.

[122] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

[123] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

[124] Reiter, E. and Dale, R. (1997). Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

[125] Ribeiro, L. F. R., Schmitt, M., Schütze, H., and Gurevych, I. (2021). Investigating pretrained language models for graph-to-text generation. In Papangelis, A., Budzianowski, P., Liu, B., Nouri, E., Rastogi, A., and Chen, Y.-N., editors, *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

[126] Ruder, S. (2018). NLP-progress - tracking progress in natural language processing. [Online; accessed 13-September-2021].

[127] Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In Màrquez, L., Callison-Burch, C., and Su, J., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

[128] Rust, P., Pfeiffer, J., Vulić, I., Ruder, S., and Gurevych, I. (2021). How good is your tokenizer? on the monolingual performance of multilingual language models. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.

[129] Salesky, E., Runge, A., Coda, A., Niehues, J., and Neubig, G. (2020). Optimizing segmentation granularity for neural machine translation. *Machine Translation*, 34(1):41–59.

[130] Saleva, J. and Lignos, C. (2021). The effectiveness of morphology-aware segmentation in low-resource neural machine translation. In Sorodoc, I.-T., Sushil, M., Takmaz, E., and Agirre, E., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 164–174, Online. Association for Computational Linguistics.

[131] Sánchez-Cartagena, V. M., Pérez-Ortiz, J. A., and Sánchez-Martínez, F. (2019). The Universitat d'alacant submissions to the English-to-Kazakh news translation task at WMT 2019. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Turchi, M., and Verspoor, K., editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 356–363, Florence, Italy. Association for Computational Linguistics.

[132] Sánchez-Cartagena, V. M., Pérez-Ortiz, J. A., and Sánchez-Martínez, F. (2020). Understanding the effects of word-level linguistic annotations in under-resourced neural machine translation. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3938–3950, Barcelona, Spain (Online). International Committee on Computational Linguistics.

[133] Schuster, M. and Nakajima, K. (2012). Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

[134] See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

[135] Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

[136] Shaham, U., Elbayad, M., Goswami, V., Levy, O., and Bhosale, S. (2023). Causes and cures for interference in multilingual translation. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15849–15863, Toronto, Canada. Association for Computational Linguistics.

[137] Sheikh, I., Vincent, E., and Illina, I. (2022). Transformer versus LSTM language models trained on uncertain ASR hypotheses in limited data scenarios. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 393–399, Marseille, France. European Language Resources Association.

[138] Shen, X., Chang, E., Su, H., Niu, C., and Klakow, D. (2020). Neural data-to-text generation via jointly learning the segmentation and correspondence. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of*

the Association for Computational Linguistics, pages 7155–7165, Online. Association for Computational Linguistics.

[139] Singh, T. J., Singh, S. R., and Sarmah, P. (2023). Subwords to word back composition for morphologically rich languages in neural machine translation. In Huang, C.-R., Harada, Y., Kim, J.-B., Chen, S., Hsu, Y.-Y., Chersoni, E., A, P., Zeng, W. H., Peng, B., Li, Y., and Li, J., editors, *Proceedings of the 37th Pacific Asia Conference on Language, Information and Computation*, pages 691–700, Hong Kong, China. Association for Computational Linguistics.

[140] Smit, P., Virpioja, S., Grönroos, S.-A., and Kurimo, M. (2014). Morfessor 2.0: Toolkit for statistical morphological segmentation. In Wintner, S., Tadić, M., and Babych, B., editors, *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 21–24, Gothenburg, Sweden. Association for Computational Linguistics.

[141] Song, H., Dabre, R., Chu, C., Kurohashi, S., and Sumita, E. (2023). Selfseg: A self-supervised sub-word segmentation method for neural machine translation. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(8).

[142] Stap, D., Niculae, V., and Monz, C. (2023). Viewing knowledge transfer in multilingual machine translation through a representational lens. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14973–14987, Singapore. Association for Computational Linguistics.

[143] Sun, Z. and Deng, Z.-H. (2018). Unsupervised neural word segmentation for Chinese via segmental language modeling. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4915–4920, Brussels, Belgium. Association for Computational Linguistics.

[144] Sutskever, I., Vinyals, O., and Le, Q. V. (2014a). Sequence to sequence learning with neural networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and

Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

[145] Sutskever, I., Vinyals, O., and Le, Q. V. (2014b). Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

[146] Taljard, E. and Bosch, S. E. (2006). A comparison of approaches to word class tagging: Disjunctively vs. conjunctively written bantu languages. *Nordic Journal of African Studies*, 15:428–442.

[147] Tay, Y., Tran, V. Q., Ruder, S., Gupta, J. P., Chung, H. W., Bahri, D., Qin, Z., Baumgartner, S., Yu, C., and Metzler, D. (2022). Charformer: Fast character transformers via gradient-based subword tokenization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

[148] Team, N., Costa-jussà, M. R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., Sun, A., Wang, S., Wenzek, G., Youngblood, A., Akula, B., Barrault, L., Gonzalez, G. M., Hansanti, P., Hoffman, J., Jarrett, S., Sadagopan, K. R., Rowe, D., Spruit, S., Tran, C., Andrews, P., Ayan, N. F., Bhosale, S., Edunov, S., Fan, A., Gao, C., Goswami, V., Guzmán, F., Koehn, P., Mourachko, A., Ropers, C., Saleem, S., Schwenk, H., and Wang, J. (2022). No language left behind: Scaling human-centered machine translation. *arXiv:2207.04672*.

[149] Tiedemann, J. (2012). Parallel data, tools and interfaces in OPUS. In Calzolari, N., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

[150] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M.,

Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv:2307.09288*.

[151] Tran, V.-K. and Nguyen, L.-M. (2018). Dual latent variable model for low-resource natural language generation in dialogue systems. In Korhonen, A. and Titov, I., editors, *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 21–30, Brussels, Belgium. Association for Computational Linguistics.

[152] Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59(236):433–460.

[153] Üstün, A. and Can, B. (2016). Unsupervised morphological segmentation using neural word embeddings. In Král, P. and Martín-Vide, C., editors, *Statistical Language and Speech Processing*, pages 43–53, Cham. Springer International Publishing.

[154] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

[155] Vedantam, R., Zitnick, C. L., and Parikh, D. (2015). Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 4566–4575. IEEE Computer Society.

[156] Vinyals, O., Fortunato, M., and Jaitly, N. (2015). Pointer networks. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

[157] Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269.

[158] Wang, C., Cho, K., and Gu, J. (2020a). Neural machine translation with byte-level subwords. In *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, pages 9154–9160. AAAI press.

[159] Wang, C., Wang, Y., Huang, P.-S., Mohamed, A., Zhou, D., and Deng, L. (2017). Sequence modeling via segmentations. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, page 3674–3683. JMLR.org.

[160] Wang, L. and Zheng, X. (2022). Unsupervised word segmentation with bi-directional neural language model. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(1).

[161] Wang, X., Ruder, S., and Neubig, G. (2021). Multi-view subword regularization. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 473–482, Online. Association for Computational Linguistics.

[162] Wang, X., Ruder, S., and Neubig, G. (2022). Expanding pretrained models to thousands more languages via lexicon-based adaptation. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 863–877, Dublin, Ireland. Association for Computational Linguistics.

[163] Wang, Z., K, K., Mayhew, S., and Roth, D. (2020b). Extending multilingual BERT to low-resource languages. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the*

Association for Computational Linguistics: EMNLP 2020, pages 2649–2656, Online. Association for Computational Linguistics.

[164] Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. (2022). Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

[165] Wiseman, S., Shieber, S., and Rush, A. (2017). Challenges in data-to-document generation. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

[166] Wiseman, S., Shieber, S., and Rush, A. (2018). Learning neural templates for text generation. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3174–3187, Brussels, Belgium. Association for Computational Linguistics.

[167] Wu, S. and Dredze, M. (2019). Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

[168] Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., and Raffel, C. (2022). ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

[169] Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., and Raffel, C. (2021). mT5: A massively multilingual pre-trained text-to-text transformer. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics:*

Human Language Technologies, pages 483–498, Online. Association for Computational Linguistics.

[170] Yang, Z., Wu, W., Yang, J., Xu, C., and Li, Z. (2019). Low-resource response generation with template prior. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1886–1897, Hong Kong, China. Association for Computational Linguistics.

[171] Yu, L., Buys, J., and Blunsom, P. (2016). Online segment to segment neural transduction. In Su, J., Duh, K., and Carreras, X., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1307–1316, Austin, Texas. Association for Computational Linguistics.

[172] Zhang, S., Dinan, E., Urbanek, J., Szlam, A., Kiela, D., and Weston, J. (2018). Personalizing dialogue agents: I have a dog, do you have pets too? In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

[173] Zhu, Y., Heinzerling, B., Vulić, I., Strube, M., Reichart, R., and Korhonen, A. (2019a). On the importance of subword information for morphological tasks in truly low-resource languages. In Bansal, M. and Villavicencio, A., editors, *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 216–226, Hong Kong, China. Association for Computational Linguistics.

[174] Zhu, Y., Vulić, I., and Korhonen, A. (2019b). A systematic study of leveraging subword information for learning word representations. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 912–932, Minneapolis, Minnesota. Association for Computational Linguistics.

[175] Zouhar, V., Meister, C., Gastaldi, J., Du, L., Vieira, T., Sachan, M., and Cotterell, R. (2023). A formal perspective on byte-pair encoding. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 598–614, Toronto, Canada. Association for Computational Linguistics.

[176] Ács, J. (2019). Exploring bert's vocabulary. [Online; accessed 13-September-2021].