# Self-Supervised Text Sentiment Transfer with Rationale Predictions and Pretrained Transformers


**Neil Sinclair**

Dissertation presented in the partial fulfilment
of the requirement for the degree of
MSc Data Science
at the University of Cape Town

**Supervisor:** Dr J. Buys

February 2022

# PLAGIARISM DECLARATION

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the required convention for citation and referencing. Each contribution to and quotation in this assignment from the work(s) of other people has been attributed, and has been cited and referenced.
3. This dissertation is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work.
5. I declare that the work contained in this dissertation, except otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this dissertation or another dissertation.

| | |
|---|---|
| SNCNEI001 | Signed by candidate |
| Student number | Signature |
| N.I. Sinclair | 01 February 2022 |
| Initials and surname | Date |

# ACKNOWLEDGEMENTS

Big thanks go out to my research supervisor, Dr. Jan Buys, for the regular check-ins, always having a paper at the tip of his fingers when I had a question about something and helping to keep me focused throughout this journey - especially at the more challenging times!

This work is based on research supported in part by the National Research Foundation of South Africa (Grant Number: 129850) and the South African Centre for High Performance Computing.

# ABSTRACT

Sentiment transfer involves changing the sentiment of a sentence, such as from a positive to negative sentiment, whilst maintaining the informational content. Whilst this challenge in the NLP research domain can be constructed as a translation problem, traditional sequence-to-sequence translation methods are inadequate due to the dearth of parallel corpora for sentiment transfer. Thus, sentiment transfer can be posed as an unsupervised learning problem where a model must learn to transfer from one sentiment to another in the absence of parallel sentences. Given that the sentiment of a sentence is often defined by a limited number of sentiment-specific words within the sentence, this problem can also be posed as a problem of identifying and altering sentiment-specific words as a means of transferring from one sentiment to another. In this dissertation we use a novel method of sentiment word identification from the interpretability literature called the method of rationales. This method identifies the words or phrases in a sentence that explain the 'rationale' for a classifier's class prediction, in this case the sentiment of a sentence. This method is then compared against a baseline heuristic sentiment word identification method. We also experiment with a pretrained encoder-decoder Transformer model, known as BART, as a method for improving upon previous sentiment transfer results. This pretrained model is fine-tuned first in an unsupervised manner as a denoising autoencoder to reconstruct sentences where sentiment words have been masked out. This fine-tuned model then generates a parallel corpus which is used to further fine-tune the final stage of the model in a self-supervised manner. Results were compared against a baseline using automatic evaluations of accuracy and BLEU score as well as human evaluations of content preservation, sentiment accuracy and sentence fluency. The results of this dissertation show that both neural network and heuristic-based methods of sentiment word identification achieve similar results across models for similar levels of sentiment word removal for the Yelp dataset. However, the heuristic approach leads to improved results with the pretrained model on the Amazon dataset. We also find that using the pretrained Transformers model improves upon the results of using the baseline LSTM trained from scratch for the Yelp dataset for all automatic metrics. The pretrained BART model scores higher across all human-evaluated outputs for both datasets, which is likely due to its larger size and pretraining corpus. These results also show a similar trade-off between content preservation and sentiment transfer accuracy as in previous research, with more favourable results on the Yelp dataset relative to the baseline.

**Key words: natural language processing, neural networks, sentiment transfer, transformers**

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVIATIONS AND/OR ACRONYMS

| | |
|---|---|
| AMT | Amazon Mechanical Turk |
| BART | Bidirectional and Auto-Regressive Transformers |
| BERT | Bidirectional Encoder Representations from Transformers |
| BLEU | Bilingual Evaluation Understudy |
| CDF | Cumulative Distribution Function |
| CHPC | Centre for High Performance Computing |
| CSIR | Council for Scientific and Industrial Research |
| DAE | Denoising Autoencoder |
| DRG | Delete Retrieve Generate paper of Li et al. 2018 |
| FGIM | Fast Gradient Iterative Modification |
| GLoVe | Global Vectors |
| GPT | Generative Pre-trained Transformer |
| GPU | Graphics Processing Unit |
| GRU | Gated Recurrent Unit |
| GYAFC | Grammarley's Yahoo Answers Formality Corpus |
| HIT | Human Intelligence Task |
| LSTM | Long-Short Term Memory |
| MLM | Masked Language Modelling |
| NLG | Natural Language Generation |
| NLP | Natural Language Processing |
| NLU | Natural Language Understanding |
| NMT | Neural Machine Translation |
| NSP | Next Sentence Prediction |
| RNN | Recurrent Neural Network |
| TFIDF | Term Frequency Inverse Document Frequency |

# CHAPTER 1

# INTRODUCTION

## 1.1 Motivation

Natural Language Understanding (NLU) and Natural Language Generation (NLG) refer to the tools and techniques used to enable computers to process and generate human readable text. These two subfields fall under the larger field of Natural Language Processing (NLP). Text is vital to our understanding of and navigation through the world as a key component of communication. Due to the importance of text in communication and its abundance in the internet-age, developing improved methods for making sense of this data is valuable for a wide range of use cases. These include text summarisation (Lewis et al., 2020; Radford et al., 2019), dialogue generation (Wolf et al., 2019), sentiment classification (Devlin et al., 2019; Radford et al., 2018; Peters et al., 2018), image description through automated captioning (You et al., 2016) language translation (Bahdanau et al., 2015; Vaswani et al., 2017), information retrieval through question and answering (Devlin et al., 2019; Peters et al., 2018; Radford et al., 2018) and novel text generation (Radford et al., 2018). NLP has grown exponentially in sophistication over the past two decades as access to textual data through the internet, coupled with the proliferation of computing power, has enabled fertile research in this field. After rule-based approaches were surpassed by predominantly statistical-based methods of generation, NLP has come to rely on neural network-based architectures as enablers for recent advances (Chernyavskiy et al., 2021). From the advent of continuous vector representations of words (Mikolov et al., 2013), to recurrent neural networks (RNNs) and Long Short-Term Memory networks (LSTMs) (Hochreiter and Schmidhuber, 1997) and, more recently the Transformer architecture (Vaswani et al., 2017), machine learning solutions continue to advance the state of the art in NLP. However, in spite of language generation that is often indistinguishable from human-written language (see Radford et al. (2019) for examples), much work is still needed in terms of controlling aspects of the text generated. This is because without having some degree of fine-grained control over what is written, generated text will have little practical value. Fine-grained control over text generation would enable one to generate text suitable for a specific audience, allowing control over tone and content, thus helping to personalise messages for audiences at scale. Therefore, research into controlling the style or sentiment of a passage of text may prove valuable

in making NLG more practical. However, controlling the style alone is not sufficient as one needs to generate text containing the desired content, and in the desired style, in order to effectively communicate.

Although some advances have been shown in the domain of conditional text generation, for example in writing text in a legal-style for a legal audience (Keskar et al., 2019), the transfer of text from one style to another still poses an open research challenge in the NLP domain. Whilst style transfer has many similarities with text translation, traditional translation methods are inadequate due to the dearth in the availability of parallel corpora in the style transfer domain. Due to this, style transfer must be treated as an unsupervised learning problem given that we do not have the target style sentence for any source sentences in our dataset. Given the unstructured nature of text data coupled with its abundance due to the internet, improving upon unsupervised and self-supervised text processing methods is a fertile area for research given the high cost of manually annotating text data. In this dissertation we explore self-supervised sentiment transfer in the domain of NLP, enabling one to generate sentences in a predetermined sentiment given the original sentence and the desired sentiment.

Along with the general goal of exploring self-supervised text processing, style transfer has a myriad of potential benefits, such as mitigating harmful content (dos Santos et al., 2018) in text, generating text in a style that would prove more receptive for an audience or, in de-formalising a piece of text (Rao and Tetreault, 2018). Sentiment transfer, as presented in this dissertation, represents a subset of the style transfer domain.

Concretely, in sentiment transfer the focus is exclusively on translating the sentiment of a text from one sentiment to another. In this domain 'translation' from one sentiment or style to another is generally referred to as 'transfer' and will be referred to as such throughout this dissertation. Here sentiment represents the attitude conveyed by the writer of a text towards the objects in the text. The sentiments in this dissertation are either positive or negative. Sentiment transfer is done by changing the sentiment of the text whilst maintaining its content. Figure 1.1 presents this graphically.

**Positive**: the food here is **amazing** and the service **excellent!**

**Negative**: the food here is **terrible** and the service **appalling!**

**Content**: the food here is … and the service …!

Figure 1.1: An example of a sentence in opposing sentiments with sentiment words in **bold** along with the content of the sentence

## 1.2 Research Objectives

The objectives of this dissertation are two fold: firstly, to examine whether a neural network-based approach to sentiment word identification can improve upon a heuristic-based approach in sentiment transfer and, secondly whether utilising a pretrained Transformer (Vaswani et al., 2017) can improve upon utilising an LSTM (Hochreiter and Schmidhuber, 1997) trained from scratch. This is done by utilising a novel method for sentence noising and a pretrained Transformer architecture for learning self-supervised sentiment transfer.

This research answers two research questions:

1. Will using the method of rationales (Lei et al., 2016; Bastings et al., 2019) improve the ability of a model to perform sentiment transfer as measured by the accuracy of the sentiment transfer and BLEU score (Papineni et al., 2002) when compared to the saliency-based noising of Li et al. (2018)?
2. Does using a pretrained language model improve upon the ability of a model to perform sentiment transfer as measured by the accuracy of the sentiment transfer and BLEU score when comparing the pretrained BART model (Lewis et al., 2020) with the model of Li et al. (2018) trained from scratch?

## 1.3 Methods

This study builds on previous work (Li et al., 2018; Sudhakar et al., 2019; Wu et al., 2019), employing a modified version of the "Delete, Retrieve, Generate" (DRG) method (Li et al., 2018). Here sentiment words are deleted and the model is trained to reconstruct the sentences in a denoising

autoencoder (Vincent et al., 2008) (DAE) fashion. Whilst the heuristic-based approach of Li et al. (2018) achieved state of the art results when published, this study utilises a new approach for identifying the sentiment-specific words to remove, namely the neural network-based method of rationales (Lei et al., 2016; Bastings et al., 2019). This unsupervised method was initially created as a method for model interpretability in NLP, however, this research shows it to be valuable for removing the sentiment words within a passage of text.

Given the rise in the popularity of the Transformer (Vaswani et al., 2017) architecture and the attention paid to transfer learning in pretrained language models, this research also compares the performance of a pretrained Transformer model, the Bidirectional and Auto-Regressive Transformer (BART) (Lewis et al., 2020) against an LSTM (Hochreiter and Schmidhuber, 1997) trained from scratch utilising only pretrained word embeddings. This model first learns to reconstruct the sentences in a DAE (Vincent et al., 2008) fashion, conditioned on either a <pos> or <neg> sentiment token, denoting the sentiment of the sentence. The trained model then generates a synthetic dataset which is used to fine-tune the model. This work is considered self-supervised because during this final part of the training, the model refines its ability to transfer from one sentiment to another utilising sentences it generated on its own.

This work utilises two datasets commonly used in sentiment transfer research (Li et al., 2018; Wu et al., 2019; Shen et al., 2017; Logeswaran et al., 2018), namely the Yelp and Amazon reviews datasets (He and McAuley, 2016). These datasets comprise short written restaurant reviews for Yelp and product reviews for Amazon that have been rated as either positive or negative. The data is explored in further detail in section 3.1.

The sentences generated by the baseline and experimental models using the two noising approaches are compared using both automatic quantitative and human evaluations. The quantitative evaluations compare the sentiment transfer accuracy and BLEU score on a set of test sentences. The human evaluations are used to enrich the automatic quantitative evaluations and further understand the efficacy of the models on these sentences. Here the human evaluators rate the sentences on sentiment transfer accuracy, fluency and content preservation. This is in line with previous research in this domain (Li et al., 2018; Wang et al., 2019; Wu et al., 2019).

## 1.4 Contributions

This research makes a number of contributions to the literature in the context of text sentiment transfer, namely:

1. It compares the relative performance of an untrained LSTM (Hochreiter and Schmidhuber, 1997) with a pretrained Transformer architecture (Vaswani et al., 2017), showing improved performance with the latter model
2. It shows an improvement in performance and fluency in the case of fine-tuning the Transformer on a new task, showcasing the benefit of transfer learning in NLP
3. It investigates the differences in results from using a simple heuristic-based method to identify words to mask or delete in a sentence versus a more sophisticated neural network-based method. This highlights how more sophisticated methods of data processing do not necessarily produce superior results
4. It proposes a method for creating a self-supervised data processing and model training pipeline. This is useful for generating a parallel training set when a corpus of unparallel data exists in multiple sentiments or styles. This method could therefore be expanded to the broader style transfer domain when altering attributes of a sentence other than its sentiment.

## 1.5 Dissertation Structure

This dissertation contains 6 chapters. Following this chapter which provides the introduction, the remaining 5 are set out in the following manner:

Chapter 2 provides an overview of key concepts in the literature that form the foundation of the research. This includes an overview of four areas in NLP, including neural network-based models in NLP, pretrained language models and transfer learning, unsupervised sentiment and style transfer, and interpretability in machine learning.

Chapter 3 provides a deeper dive into the methods employed in this dissertation for conducting the research. This starts with a discussion of the datasets used in this dissertation. This is then followed by a detailed explanation of the method of rationales (Bastings et al., 2019) as well as the saliency method used as a baseline for sentence noising. We then provide detail on the pretrained BART

(Lewis et al., 2020) model used as well as the Delete Retrieve Generate (DRG) (Li et al., 2018) DeleteOnly model used as a baseline. Finally, we outline the training pipeline used for teaching BART to transfer from one sentiment to another.

Chapter 4 provides a discussion of the resources used in training. This is followed by a detailed overview of how each of the models in the dissertation are set up and run, including the hyper-parameters used. We then provide an explanation of the quantitative and qualitative evaluation metrics used to compare our results with the baseline. Finally, chapter 4 ends with a discussion of the training curves seen for the BART models.

Chapter 5 provides a detailed discussion of the results, comparing the BART and DeleteOnly models as well as the rationales and saliency noising methods. This is done by comparing the results on the test set utilising both quantitative and human evaluations. After a discussion of the results, the two research questions are answered.

Chapter 6 provides a summary of the research, discusses potential shortcomings and presents recommendations for future research.

# CHAPTER 2

# LITERATURE REVIEW

This chapter provides an overview of key research in the field of neural network-based NLU and NLG. For simplicity, NLU and NLG are referred to singularly as NLP going forward. We begin with an overview of neural network-based NLP models before providing a detailed discussion of the Transformer (Vaswani et al., 2017) architecture. We then focus on the benefit of pretrained models and the advantage of transfer learning in NLP. This is followed by a discussion of non-parallel sentiment and style transfer in NLP before finally providing a brief overview of interpretability in neural networks. The information in this chapter lays the foundation for the following chapter which describes how these models are implemented in this dissertation.

## 2.1 Neural Network-Based Language Models

### 2.1.1 Conditional Language Distribution

A language model is a statistical model which learns the distribution of words in a language. One of the key challenges with NLP is the sequential nature of language. That is, order matters and words follow in a manner that is related to the words that come before them. The way that this is encoded in probabilistic models is through conditional probability. Utilising conditional probability and the chain rule of probability, the learning problem can be conceptualised as learning the probability of word$_i$ given the words preceding it. The probability of a sentence or document can thus be represented by the product of these conditional probabilities and is defined as

$$\begin{aligned} P(X) &= P(x_1)P(x_2|x_1)P(x_3|x_2,x_1)...P(x_T|x_{<T}) \\ &= \prod_{i=1}^{T} P(x_i|x_{i-1},x_{i-2},...,x_1), \end{aligned} \tag{2.1}$$

where $T$ refers to the length of document being processed, $X$ refers to the document and $x_i$ represents word$_i$ in the document. Using the log rule, the log probability of document $X$ can be decomposed as the sum of the log probabilities of each word conditioned on the words that come

before and is defined as

$$\log(P(X)) = \sum_{i=1}^{T} \log P(x_i|x_{i-1}, x_{i-2}, ..., x_1), \tag{2.2}$$

### 2.1.2 Word Vectors

Although many advances have been achieved in NLP utilising simplifying assumptions, such as treating a document as simply a "Bag of Words" where word order was not important, two key advances removed this simplifying assumption. The first of these was the creation of continuous distributed vector representations of words such as Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014). By utilising a neural network architecture and learning a set of weights that were utilised to predict whether a word was present within a window from another word, this method learns contextual representations known as word embeddings. Most famously, these contextual representations of words allow one to conduct vector arithmetic on these learned representations where a linear relationship exists that can be conveyed as *Man is to king as woman is to queen* (Pennington et al., 2014).

Concretely, these contextual representations learn a reduced dimensional space for words where similar words are close to each other. The above example suggests that the relation between man and king is similar to the relation between woman and queen and that this could be expressed with a vector. Mathematically, this can be expressed as

$$\begin{aligned} \vec{king} - \vec{man} &= \vec{x} \\ \vec{woman} + \vec{x} &= \vec{queen}, \end{aligned} \tag{2.3}$$

where the vector $\vec{x}$ can be seen to convey the semantic concept of *royalty*. These continuous contextual representations have subsequently played a significant role in advancing NLP, both in practice and research.

### 2.1.3 Recurrent Neural Networks

The second fundamental advancement in NLP was the advent of the Recurrent Neural Network (RNN). This architecture considers the temporal aspect of information, such as language, by processing the data one time-step at a time with the data at each time step including a representation of the data from all of the previous time steps. At each time step the current data, such as a word token represented by a Word2Vec (Mikolov et al., 2013) vector, is processed along with the representation of all of the words that have come before it.

In spite of this improvement, vanilla RNNs experience the challenge of exploding or vanishing gradients (Hochreiter and Schmidhuber, 1997) whereby the gradients for learning become unstable during training as they backpropagate through time in the network. This causes learning not to be feasible. To combat this, the LSTM (Hochreiter and Schmidhuber, 1997) architecture was created. Through a set of memory cell and gate units that mitigate the possibility of gradients vanishing or exploding, this architecture enables learning over many discrete time steps. This enables the creation of rich representations of sentences that have multiple downstream uses, such as text generation and sentence classification.

### 2.1.4 Attention Mechanism

Initially, RNNs were limited by having to encode a document into a single hidden state, however, an additional advancement in NLP has enabled neural network-based models to differentially weight tokens at different time steps within a document. This has been termed the attention mechanism (Bahdanau et al., 2015; Luong et al., 2015) as it allows a model to give a different attention weighting to different parts of a document. This mechanism was initially developed within the context of natural language translation and utilised by an encoder-decoder architecture (Sutskever et al., 2014; Cho et al., 2014) that was trained to translate from one language to another. In the original paper Bahdanau et al. (2015) showed how learning of a soft attention over the hidden states at each time step in the encoder enables the decoder to pay selective attention to words in the encoder. This proved to be highly effective when producing a translation from one language to another. Prior to this, the decoder part of the network was conditioned on a pooled hidden representation of all of the information from the encoder in the network, represented by the final

hidden state of the encoder. This token-level attention removed the heavy reliance on the final hidden state of the encoder to attempt to effectively capture all of the relevant information in a sentence and enabled state of the art results in machine translation (Bahdanau et al., 2015). Intuitively, one can understand how the step-by-step generation of a translation is likely to be most influenced by only a few words in the source sentence at once. Each word in the translation would be dependent predominantly on one or two words in the source sentence. Attention can be summarised as the mechanism that allows the decoder to selectively attend to specific parts of the source sentence when generating the target sentence.

### 2.1.5 The Transformer

Although the RNN through the LSTM (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Cho et al., 2014) variations of the architecture managed to achieve state of the art results in NLP for many years, the most recent advances in neural network-based language models have been driven by the Transformer (Vaswani et al., 2017) architecture. This architecture improves upon previous recurrent architectures by enabling the model to process tokens in parallel through the implementation of a self-attention mechanism. This self-attention mechanism is similar to the attention mechanism proposed by Bahdanau et al. (2015) and Luong et al. (2015), however, it is used to represent the relation of each token to every other token within a document. By applying attention over the entire sequence in a single computation, albeit over a number of layers, this reduces the time burden of having to process a sentence sequentially, one token at a time. As noted by Vaswani et al. (2017), whilst a recurrent layer requires $O(n)$ sequential operations, self-attention requires only $O(1)$ sequential operations and is faster when the sequence length $n$ is smaller than the size of the sequence dimensionality, $d$. In the case of the original Transformer the dimensionality of the model and self-attention is $d_{model} = 512$. Given that the self-attention mechanism enables the model to learn the relationship of every token to every other token in a sentence, it also leads to improved handling of long-term dependencies between tokens vis-à-vis earlier systems (Vaswani et al., 2017).

Figure 2.1 provides a visual representation of the original architecture of the Transformer comprising both encoder and decoder stacks.

The encoder, shown on the left of Figure 2.1, consists of six identical stacked sub-layers following a

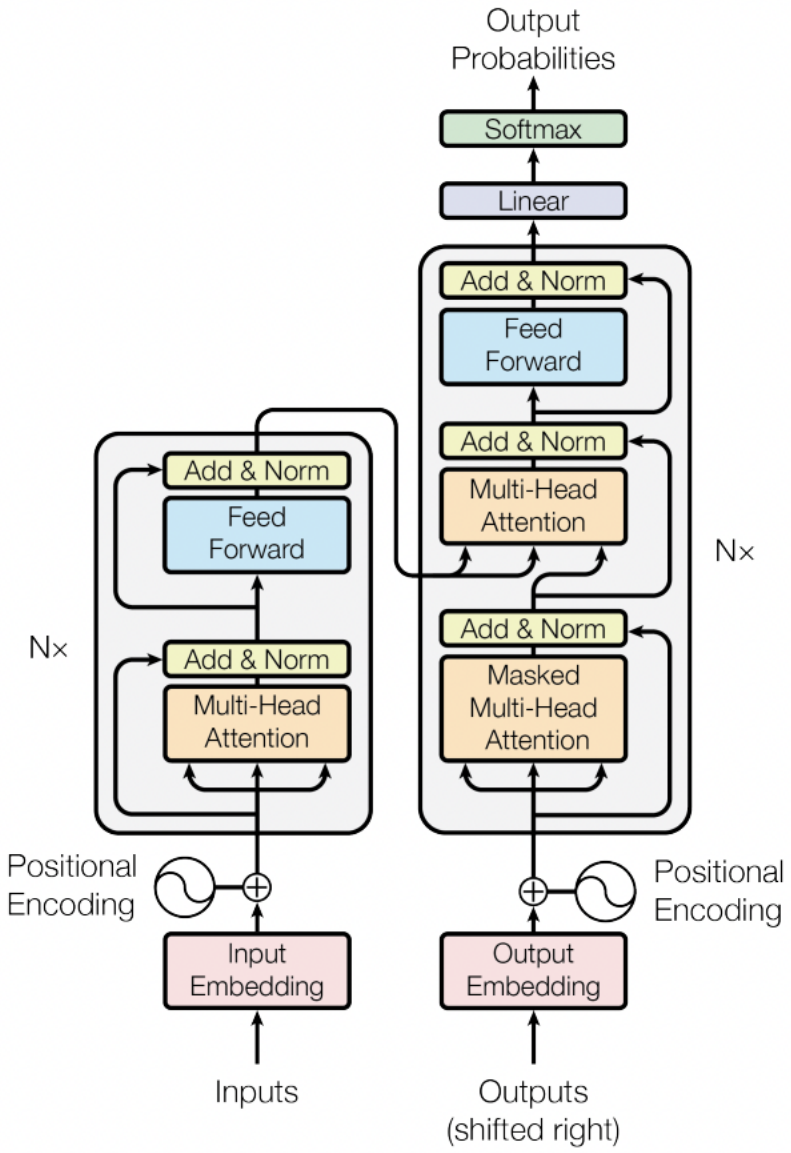


Figure 2.1: The Transformer architecture from Vaswani et al., 2017. The left hand side shows an example of the encoder block with sub-layers and the right hand side shows an example of the decoder block with sub-layers.

token and positional embedding layer. Each of the sub-layers includes a set of eight self-attention heads, employing the self-attention mechanism over the source sentence encoding. This is shown as the multi-headed attention sub-layer in Figure 2.1. Each sub-layer takes as input the previous sub-layer, applies self-attention to this input and then passes this forward to a position-wise fully

connected layer. This processed output is added to the output from the previous sub-layer with a residual connection and then passed through a batch normalisation layer before being passed to the next layer.

The decoder layer, shown on the right of Figure 2.1, is similarly constructed of 6 sub-layers. However, unlike the encoder stack, the decoder employs masked multi-headed attention in the first layer of each sub-layer such that positions cannot attend to subsequent positions. This prevents the model from trying to look forward when decoding a piece of text. This first masked self-attention sub-layer is then followed by a self-attention sub-layer over the output from this first sub-layer, as well as the output from the corresponding encoder layer. This is represented by the output arrow going from the encoder layer to the decoder layer in Figure 2.1. This combined output is then fed to a simple position-wise feed forward layer and added to a residual layer before going through layer normalisation.

The self-attention mechanism of the Transformer employs Scaled Dot-Product Attention, utilising a set of Queries ($Q$), Keys ($K$) and Values ($V$), which are explained in more detail below. This scaled attention enables the model to pass forward an attention-weighted version of the Value vector inside each attention sub-layer as follows

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_2}})V. \tag{2.4}$$

The self-attention mechanism of the Transformer employs multi-headed attention. This is done by utilising $h$ attention heads over a set of $h$ $Q$, $K$ and $V$ vectors. These vectors comprise projections of the original vectors of size $\mathbb{R}^{d_{model}}$ to a lower dimensional space before being passed through the scaled dot-product attention of the model represented by Equation 2.4. Finally, after the vectors have been processed through the multi-headed attention, the outputs are concatenated and projected back into the original $\mathbb{R}^{d_{model}}$ vector space. For clarity, the multi-headed attention is represented as

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, ...., \text{head}_h)W^O \\ \text{where head}_i &= \text{Attention}(WQ_i^Q, KW_i^K, VW_i^V), \end{aligned} \tag{2.5}$$

here the projections are controlled by the parameter matrices, $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}, W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}, W^O \in \mathbb{R}^{hd_v \times d_{model}}$.

The role of the $Q$, $K$ and $V$ vectors can better be understood in the following manner. Within the encoder, all of the $Q$, $K$ and $V$ vectors come from the same place, namely the output of the previous layer of the encoder. It is in this way that each position in the encoder can attend to every position from the previous layer in the encoder, building rich contextual representations of sentences.

In the decoder, the queries come from the previous layer of the decoder, and keys and values come from the output of the encoder. This allows the decoder to attend to every position of the input, similarly to the traditional attention mechanism proposed by Bahdanau et al. (2015) and Luong et al. (2015). Note, however, that in the decoder's self-attention layer, the masking within the attention ensures that each position can only attend to positions up to and including its position.

Prior to being fed to the next layer, the output of the multi-headed attention is fed through two fully connected feed-forward layers, shown as the blue sub-layers in Figure 2.1. The fully connected layers attend to each position separately and identically making their operation similar to that of two 1D convolution filters with kernel size one. The parameters within each layer are shared, but are different between layers. In the original Transformers architecture, the fully connected layers project the $d_{model} = 512$ output of the multi-headed attention to a dimensionality of $d_{ff} = 2048$ before projecting back down to the original $d_{model}$ dimensionality.

Due to the parallel processing employed by the self-attention layers, it is necessary to include a positional embedding in the input to the Transformer along with the usual learned token embeddings. This positional encoding is shown prior to the first blocks in both the encoder and decoder in Figure 2.1. The positional embedding chosen for the original Transformer model was fixed, however, a learned embedding can similarly be used. Prior to the positional embedding, the tokens are fed to the model through the input and output embedding layer shown in Figure 2.1. The token embedding layer of the original Transformer is similar to the token embeddings learned from Word2Vec (Mikolov et al., 2013) in that each word is represented by a continuous vector. In the Transformer architecture, the input and output embeddings are tied as per Press and Wolf (2017) as this has shown improvements in the quality of language models.

The first version of the Transformer achieved state of the art results in a machine translation task using a sequence-to-sequence encoder-decoder model (Vaswani et al., 2017). Subsequently, many variations of the original transformer architecture have been conceptualised and experimented with, achieving state of the art results in a variety of tasks. These include dialogue generation (Wolf et al., 2019; Zhang et al., 2020), question answering (Radford et al., 2018) and summarisation (Lewis et al., 2020). Two of the most notable adaptations to the original Transformer architecture have been BERT (Devlin et al., 2019) and GPT (Radford et al., 2018). BERT comprises solely of the encoder part of the Transformer architecture and is optimised through a masked-language-modelling (MLM) objective where parts of a sentence are masked and the model must learn to fill in the masked words (Devlin et al., 2019). GPT on the other hand comprises solely of the decoder part of the Transformer architecture and is optimised through a next-word prediction task (Radford et al., 2018). Both of these architectures achieved state of the art results on a variety of downstream tasks with minimal fine-tuning and are covered in greater detail in the following section.

Whilst BERT (Devlin et al., 2019) and GPT (Radford et al., 2018) and later variations of these models utilised a simplified Transformer architecture, utilising only the encoder or decoder stack of the overall architecture, recent approaches are again utilising the connected encoder-decoder architecture (Lewis et al., 2020; Keskar et al., 2019; Raffel et al., 2020).

## 2.2 Pretrained Language Models and Transfer Learning in NLP

Transfer learning refers to the process of taking a model trained in one domain and transferring its learning to another domain with very little or no fine-tuning (Ruder, 2017). The key result of pretraining language models and using them for downstream tasks is the ability to utilise transfer learning in the NLP domain. Thus, this dissertation explores the impact of using transfer learning in sentiment transfer to enhance performance on the task when compared to models trained from scratch particularly for the problem.

### 2.2.1 Transformer Pretraining

As noted in subsection 2.1.1 of this chapter, a language model is a statistical model that learns the conditional distribution of words in a language. Pretraining in the context of NLP refers to training a language model on a language modelling task in an unsupervised manner using unlabelled text

data. Current unsupervised approaches comprise training the language model with either a next-word prediction language modelling task (Radford et al., 2018) or with a denoising autoencoder-like task of MLM (Devlin et al., 2019). In the case of the auto-regressive next-word prediction task as with GPT (Radford et al., 2018), the model is optimised by learning the conditional distribution of words given the preceding words. This is implemented by optimising the negative log-likelihood of the probabilities of the next word prediction over the actual label as described in Equation 2.2. As noted in Radford et al. (2018), the probability of the next token is calculated first by processing the tokens through each successive transformer block and then applying a softmax on the final layer. This successive process is defined as

$$\begin{aligned} h_0 &= UW_e + W_p \\ h_l &= \text{transformer\_block}(h_{l-1}) \ \forall i \in [1, n], \end{aligned} \tag{2.6}$$

where $U$ is the set of context or conditioning tokens, $W_e$ and $W_p$ represent the word and positional embedding matrices respectively, $n$ is the number of sub-layers with $h_n$ representing the final layer and $W_e^i$ representing the word embedding for the $i^{th}$ token. The final softmax applied to the last layer outputs the probability that the next word $X = x_i$ which is represented as

$$\begin{aligned} P(X = x_i) &= \text{softmax}(h_n W_e^T) \\ &= \frac{e^{h_n W_e^i}}{\sum_k e^{h_n W_e^k}}. \end{aligned} \tag{2.7}$$

Finally, the model is optimised by calculating the negative log-likelihood of this vector of output probabilities of length $|V|$, the length of the vocabulary, with a one-hot-encoding of the actual next word in the corpus. This loss is defined as

$$\text{L} = -\frac{1}{N} \sum_i^N y_i^T \log\left(\text{softmax}(h_n W_e^T)\right), \tag{2.8}$$

where $N$ represents the length of the training batch and $y_i$ represents a one-hot-vector, which is the length of the vocabulary, for word $i$. This one-hot-vector has a 1 in the position of the correct word in the vocabulary and a 0 everywhere else.

In order for words to be fed into the model, they must first be tokenized. Concretely, this involves splitting a sentence up into a set of constituent tokens such that a sentence like **The cat sat on the mat** may be broken up into a set of seven tokens, **'The', 'cat', 'sat', 'on', 'the', 'mat', '.'**. Here it is noted that the sentence is split into its *constituent tokens* because the complexity of the tokens lies on a spectrum from character-level tokens to word-level tokens, depending on the researcher's design decision.

Originally developed in order to compensate for the occurrence of rare words in neural machine translation (NMT) two primary methods are usually employed to tokenize the text for Transformer models, namely Byte Piece Encoding (BPE) (Sennrich et al., 2016) and WordPiece Encoding (Wu et al., 2016). The reason that rare words pose a problem for NMT is that the size of the vocabulary of language models must be fixed, however, given the occurance of rare words, having a fixed vocabulary that could handle all words in a language would be computationally infeasible. BPE (Sennrich et al., 2016) and WordPiece (Wu et al., 2016) alleviate this problem by splitting words up into smaller, commonly occurring parts. In the case of Sennrich et al. (2016) the authors employ the Byte Pair Encoding algorithm of Gage (1994) which iteratively replaces the most frequently occurring pairs of bytes with a single unused byte. In the case of language, this could be implemented by replacing the bytes for 't' and 'h' with the single byte 'th' given that the n-gram 'th' is one of the most frequently occurring in the text. Each new merge operation then produces a new symbol representing a character n-gram (Sennrich et al., 2016) until a stopping point is reached. The method employed by Wu et al. (2016) for WordPiece is similar to that used in BPE (Sennrich et al., 2016), with both of these methods achieving a balance between the efficiency of whole-word tokenization and the flexibility of character-level tokenization.

In MLM a number of the tokens are replaced with a <mask> token and the model is optimised to predict the correct word in the place of the <mask> token. Concretely, the original and masked document would look like the example in Figure 2.2.

Similar to the auto-regressive generation loss shown in Equation 2.8, the model learns to minimise

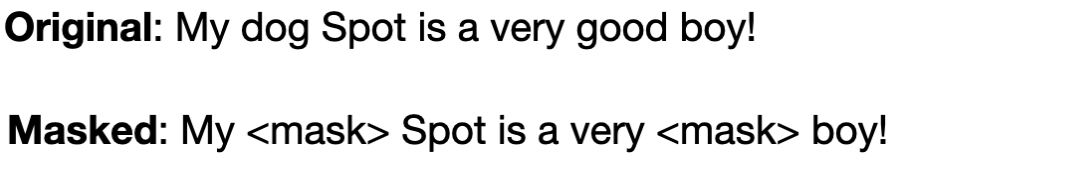


Figure 2.2: An example of an original sentence and the corresponding masked sentence for MLM

the cross-entropy loss, but only on the masked words. This loss is defined as

$$\mathrm{L} = -\frac{1}{N}\sum_{i}^{N}\sum_{j}^{M} y^T \log\left(\mathrm{softmax}(h_n W_e^T)\right), \tag{2.9}$$

where $i$ represents the index of sentence $i$ from a training set size of $N$ and $j$ refers to the $j^{th}$ masked token of the $M$ masked tokens in sentence $i$.

In the case of Bidirectional Encoder Representations from Transformers - or BERT (Devlin et al., 2019) - the model is also trained on a next-sentence prediction (NSP) task in a semi-supervised fashion. This is based on the understanding that certain downstream tasks like Question Answering and Natural Language Inference require understanding of the relationship between two sentences. A semi-supervised training set is created by constructing positive and negative next-sentence training pairs. This is done by concatenating sentences with positive (actual) next sentences as well as with incorrect next sentences and labelling the data as IsNext and NotNext respectively. Here Devlin et al. (2019) use a special token appended to the beginning of the sentence, [CLS], to feed into the classifier for the NSP task. In addition to the [CLS] token, BERT also utilises segment embeddings for the NSP task to identify the first and second sentence of the sentence pairs. These learned embeddings are added to the token and position-wise embeddings, similar to those used in the original Transformer model.

One of the key differences between BERT and GPT is that, as the name suggests, BERT utilises a bidirectional representation of the text fed into the model. Comprising solely of the "encoder" block of the Transformer architecture, the multi-headed attention employed in BERT is applied to the entire input at once, thus allowing each token to attend to tokens both before and after its current position. Note that this is in contrast to GPT where the multi-headed attention limits

tokens to attend only to preceding tokens.

For the language modelling tasks, models that achieved state of the art results at the time of their publishing were trained on massive datasets. For example GPT was trained on 7,000 books (Radford et al., 2018), GPT-2 (Radford et al., 2019) on 8-million documents comprising 40GB of text, BERT (Devlin et al., 2019) on two corpora comprising a total of 3,300M words and lastly, the T5 (Raffel et al., 2020) model on a corpus comprising 750GB of text. After training on either the auto-regressive next-word prediction or MLM task, pretrained language models have been shown to achieve state of the art results on a series of down-stream NLP tasks with minimal fine-tuning (Radford et al., 2018). This also extends to the case, as with BERT, where the final output of the model is fed to a linear classification layer which is trained independently of the pretrained model in a very short space of time (Devlin et al., 2019). In cases where an additional linear layer is added to the pretrained models, the pretraining of the model has learned a rich latent representation of the document(s) it processes which are optimised for downstream tasks. These models have also been shown to achieve strong results with few-shot and even-zero-shot learning on a number of tasks (Radford et al., 2019).

The approach of pretraining a language model on a general language modelling task prior to fine-tuning on a downstream task was first employed by Howard and Ruder (2018); Peters et al. (2018). In this work, the authors pre-trained LSTM (Hochreiter and Schmidhuber, 1997) models on a language modelling objective and then fine-tuned the pretrained model on a classification task. Both Radford et al. (2018) and Devlin et al. (2019) furthered this work by using the Transformer architecture and by fine-tuning the pretrained model on a wider variety of tasks, in turn achieving results at or close to state of the art. Subsequent to the initial work by Radford et al. (2018), a larger model, GPT-2 (Radford et al., 2019) was trained that could match a number of state of the art baselines on a variety of NLP tasks simply by conditioning the output generated by the model. For example, after pretraining, GPT-2 was able to answer a noteworthy percentage of question answer pairs simply by conditioning the generation of text with a set of question and answering pairs. This is an example of zero-shot learning where the model was able to achieve favourable results in a task it was not explicitly trained for and without any task-specific fine-tuning. GPT-2 was also able to summarise text when conditioned on the original text concatenated with the "TL;DR" term which, expressing the idea "Too Long; Didn't Read", is commonly used to represent a summary for text

on the internet. Whilst GPT has shown favourable results in a number of tasks, the denoising language modelling objective of BERT (Devlin et al., 2019) has been shown to provide improved results for downstream tasks when the final output of the model is combined with a fine-tuned linear classification layer. Similarly, the T5 model (Raffel et al., 2020) has been trained such that it treats all problems as 'text-to-text' and thus is essentially conditioned to complete a series of tasks, such as question and answering or translation, depending on a text input. Along with a host of "standard" NLP tasks, pretrained models can also be fine-tuned on novel conditional language generation tasks, as has been done in the case of dialogue generation (Wolf et al., 2019; Zhang et al., 2020).

### 2.2.2 Transfer Learning

Transfer learning has had an enormous impact in the computer vision space, where models trained on the generic ImageNet dataset could be adjusted slightly and fine-tuned with minimal effort to work on different image classification tasks with different datasets (Ruder, 2017). Arguably, computer vision models learn rich representations of the images they process that can readily be used in slightly different domains from what they were trained on. In the case of NLP, results from using pretrained models suggest that these models capture deep semantic information in language as they encode the relationships between words. This deep semantic information can then be used by downstream layers in the models to transfer the learning of these models to novel situations. This is exemplified where the outputs of a BERT model are fed to an additional linear classification layer during fine-tuning for a classification task (Devlin et al., 2019). Whilst the BERT model's weights are frozen and only the classification layer(s) are trained, the model manages to transfer the learning from a denoising language modelling and next sentence prediction domain to a classification domain with only minimal fine-tuning.

The richness of the information captured by these pretrained language models can be seen, for example, in abstractive summarisation tasks where background information not included in the source text is generated in the summary text (Lewis et al., 2020). Through the ability to fine-tune these models on novel tasks, such as dialogue generation (Wolf et al., 2019; Zhang et al., 2020), question-and-answering or classification (Radford et al., 2018; Devlin et al., 2019), these models are arguably leveraging the rich language representations learned and simultaneously leading to a

reduction in the cost of utilising language models.

## 2.3 Non-Parallel Sentiment and Style Transfer in Natural Language Processing

### 2.3.1 Conditional Text Generation

Generation of text in a certain style can be posed as generating text conditioned on a style marker. Conditional generation has been employed in a number of domains, for example, in the few- and zero-shot learning employed by GPT-2 as noted in subsection 2.2.1. Furthermore, Keskar et al. (2019) curated a training set such that the usual language modelling objective of their model would be conditioned on the domain from which the text came. The domain here was defined by where the data was pulled from the internet. For example, when the model was learning to generate text in the style of a legal document or news report, a token indicating the style of the text was included during next word prediction during training. This allows the model to generate text in a certain style conditioned on a style token and a short seed phrase.

Although conditional text generation has shown promising results, as with the CTRL model (Keskar et al., 2019), this conditional generation lacks the ability to control the content as well as the style of the sentences being generated. In comparison, one of the advantages of style transfer is that it allows one to condition the generation of a sentence on both the desired style, as well as the desired content, of the target sentence.

### 2.3.2 Overview of Sentiment and Style Transfer

Style transfer refers to the process of transferring the style of a piece of text from one style to another whilst preserving the style-independent content of the text (Li et al., 2018). As noted in the introduction, sentiment transfer, as explored in this dissertation, is a subset of the style transfer domain. As an example of style transfer, sentiment transfer is shown in Figure 2.3 with the transferring of a sentence from the positive to negative sentiment.

Additional examples include transferring the style of a sentence from male to female and old to young (Subramanian et al., 2018) as well as formal to informal (Rao and Tetreault, 2018). In the context of style transfer, sentences can be seen as a combination of content and style elements, where the content is defined as the information not captured by the style elements and where

**Positive**: the food here is **amazing** and the service **excellent!**

**Negative**: the food here is **terrible** and the service **appalling!**

Figure 2.3: An example of a sentence in two sentiments with sentiment words in **bold**

the goal of style transfer is to alter the style whilst maintaining the content (Logeswaran et al., 2018). To this end, a variety of both human and automatic evaluation metrics have been used together. These include the automatic BLEU (Papineni et al., 2002) and accuracy metrics as well as human evaluations on the style, content preservation and grammatical-correctness, or fluency, of the sentences transferred from one sentiment to another (Li et al., 2018; Wu et al., 2019). These metrics are covered in greater detail in section 4.3 of this dissertation.

One of the key challenges of implementing style transfer in the NLP domain is that for almost all corpora, matched sentences across different styles are not readily available (Sudhakar et al., 2019). Although in very few cases parallel corpora do exist – such as in the human annotated GYFAC corpus of Rao and Tetreault (2018) – this discussion will exclude supervised methods of style transfer. The non-parallel nature of style corpora indicates the need to treat this as an unsupervised learning problem where the style of the source text is known, however, target examples are not available. Although there are two overarching methods currently employed in the literature for NLP style transfer, which are outlined below, many of the sub-methods within both of these overarching methods employ a DAE (Fu et al., 2018; Vincent et al., 2008) objective. This method essentially learns to reconstruct a noised sentence in an unsupervised manner, where a model learns to utilise a style token added to a sentence to correctly reconstruct the sentence. This is intended as a solution for the unsupervised approach required in this domain. This is explored in greater detail in the Methods chapter.

Building on the work of Radford et al. (2019) where GPT-2 was able to show noteworthy results with conditional language generation, including zero-shot learning, CTRL (Keskar et al., 2019) was developed to further refine conditional text generation. This model was designed to generate text in a specific style, conditioned on a style control token pre-pended to the beginning of a sentence whilst the model was trained with an auto-regressive text generation task. The style token was included during training, where the training corpus was split up into subsets, each mapped to a

specific style token. This model has shown impressive results with regards to generating text in a specific style, however, it has not been trained to transfer text from one style to another as it is solely used for conditional text generation.

### 2.3.3 Delete Retrieve Generate Approach

As noted previously, there are two overarching methods for style transfer in NLP. The first is a method which can be categorised as 'Delete, Retrieve and Generate' (DRG) which comprises finding and deleting style-specific words, and then generating a new sentence conditioned on a style attribute token - such as "positive". Li et al. (2018) use a simple saliency-based method for locating which style words to remove from a sentence, thereby creating a sentence comprising only content words. This sentence can said to be 'noised' as it represents a noisy version of the original sentence given that the style words have been removed. The authors then use these noised sentences with a DAE objective to reconstruct them during training. Finally, at test time they swap the original style token of a noised sentence with its opposite and generate a sentence in the new style conditioned on this new style token and the sentence content. This is referred to as the DeleteOnly version of the DRG model. Sudhakar et al. (2019) follow a similar approach, however, the authors use a Transformer model, utilising the attention weights of the model to identify which words to remove. By conditioning on the content of the original sentence and combined with the style words of the closest matching sentence in the corpus of the desired style, the model is able to transfer from one style to another. Here closest matching refers to two sentences where the content words are most similar. Wu et al. (2019) also utilise a Transformer model, specifically using a pretrained BERT (Devlin et al., 2019) model. The authors use a combination of n-gram saliency and attention weights as approaches to identify which style words to remove from the source sentences, building a vocabulary of style words to remove in the corpus. They then use a combination of DAE-like reconstruction and discriminator-led style transfer to train their model to generate sentences in a new style. In all of the examples that utilise variations of the DRG method, the specific goal within the domain of style transfer has been sentiment transfer.

### 2.3.4 Latent Representation Approach

The second approach involves encoding sentences in one style, manipulating the encoded latent representation of the sentences and then using a decoder to generate the sentences in another style. Within this approach a number of sub-approaches can be identified. He et al. (2020) approach style transfer from a variational inference perspective where the authors treat the problem as inferring a set of latent variables from which the examples in the corpus are generated. Through a combination of generating sentences in one style from this latent distribution and then using back-translation to reconstruct the sentences, the authors train a model that achieves close to state-of-the-art results on a number of tasks. Logeswaran et al. (2018) approach style transfer as a combination of learning an autoencoder (reconstruction) and back-translation objective, coupled with an adversarial objective to ensure the generation of realistic sentences. Here the authors learn to control multiple attributes of the style of a sentence simultaneously during training. Wang et al. (2019) utilise a Transformer to learn an entangled latent representation of a sentence (i.e. one that comprises both style and content) and then use a pretrained classifier to edit the entangled latent representation of the sentence. This method transfers the original sentence from the source to target style by finding the latent representation closest to the original but conforming to the desired style. They then use an RNN decoder to reconstruct the sentence in the desired style. Subramanian et al. (2018) treat style transfer more simply as the combination of an autoencoding and back-translation objective, doing away with the adversarial methods used in the papers previously.

### 2.3.5 Comparison of Approaches

The research by Wu et al. (2019) achieved the highest accuracy results on the Yelp dataset (He and McAuley, 2016) when compared with other models (Li et al., 2018; Sudhakar et al., 2019; Wang et al., 2019) that utilise this dataset across both high level methods outlined above. This may partially be due to the objective of the BERT-based model used in the research being simply to fill in the missing words. Whilst other models that delete words and then generate new sentences (Li et al., 2018; Sudhakar et al., 2019) must generate the entire sentence conditioned on the content and the style token, the model of Wu et al. (2019) only focuses on replacing the missing style words, thus focusing and simplifying the training objective. This model may have also outperformed Wang et al. (2019) due to the Yelp dataset being well suited for style transfer with style word identification

and alteration. This is explored further when discussing the datasets. Thus, although the "Mask and Infill" model (Wu et al., 2019) performed well, its use is likely to be limited to problems that can be narrowly defined as removing and replacing words to change the style of a sentence.

The research by Wang et al. (2019) achieved the highest BLEU score with the second highest accuracy score for style transfer on the Yelp dataset. They also achieved the highest accuracy and BLEU score for the Amazon dataset. As noted previously, a classifier edits the latent space during style transfer. One advantage of the method used is that a weighting parameter enables one to actively change the degree of the sentiment in the style transfer. Their method also enables sentiment transfer over multiple style attributes at once. Whilst Subramanian et al. (2018) do not incorporate an adversarial loss in their training, focusing instead on back-translation, the work by Wang et al. (2019), which does incorporate an adversarial loss and latent space editing, achieves higher accuracy and BLEU scores. This suggests that operating directly on the latent space during training may be beneficial in style transfer. This method, which is also not reliant on replacing specific style words, may have wider applicability in style transfer when compared to the DRG methods (Li et al., 2018; Sudhakar et al., 2019; Wu et al., 2019), such as the one used in this dissertation, where the style of a sentence may be dependent on more than just a few key words.

## 2.4 Interpretability in Natural Language Processing

Neural network models are generally considered as 'black box' models because, compared to other methods such as linear regression or vanilla decision tree methods, it is challenging to understand how these models process features to arrive at their predictions (Lei et al., 2016). Given the increasing ubiquity of neural network-based models in many areas of our lives, numerous authors have argued for the necessity of making these models more interpretable (Lei et al., 2016; Lundberg and Lee, 2017). It is argued that understanding how models arrive at decisions will not only help us to understand their reasoning, but also help researchers to improve models (Sundararajan et al., 2017).

Interpretability is explored in this dissertation because it is seen to have applications outside of solely understanding how a model derives its predictions. The representations learned through interpretability can be used for additional use-cases, and in the case of this dissertation, interpretability

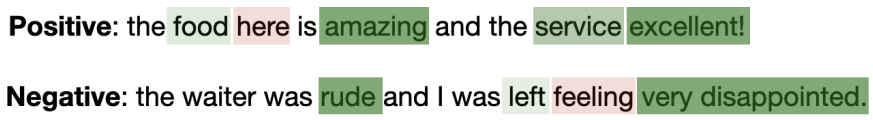


Figure 2.4: An example of the attributions given to different words based on how they contribute to the sentence classification using Integrated Gradients. Green represents a positive attribution and red a negative, with the shade indicating the magnitude of attribution.

is used as a tool for identifying which sentiment words to mask or delete in a sentence.

### 2.4.1 Integrated Gradients

A number of promising areas in neural network interpretability have emerged in recent years. Gradient-based methods aim to understand which parts of the input are sensitive to changes in the output, thus allowing one to understand the importance of features, or more broadly, allowing one to understand the rules the model uses to make decisions. A method known as Integrated Gradients (Sundararajan et al., 2017) has been designed to meet a set of axiomatic criteria that the authors consider essential for making statements about attributions in a neural network. These axiomatic criteria have been constructed to mitigate shortcomings with previous methods. Integrated Gradients essentially sums the gradients for each feature as the input is linearly interpolated from a baseline input - such as the <PAD> token embedding for text or blank image for an image - to the original input (Sundararajan et al., 2017). In this manner it uncovers the contribution of each feature to the output. This method has been applied to a range of problems, including computer vision and NLP. Integrated Gradients can be applied to a neural network without any architectural adjustments.

Figure 2.4 provides a graphical example of how integrated gradients can be visualised. Here we see that words that have a stronger positive attribution with a sentence classification prediction would have a strong positive score, indicated by a deeper green shading. Words with a negative contribution to the prediction would be shaded in red and neutral words would not be shaded or would be shaded very lightly.

### 2.4.2 DeepLift

A similar method to integrated gradients comprises the method of DeepLift (Shrikumar et al., 2017). DeepLift does not require the calculation of gradients, thus being arguably more computationally efficient than Integrated Gradients. However, both DeepLift (Shrikumar et al., 2017) and Integrated Gradients (Sundararajan et al., 2017) calculate the attribution of the input utilising a reference input. Both methods then utilise the difference between the actual and reference values of the input to calculate the attribution of each data point in the input. Concretely, in both cases the authors utilise the difference between zero-valued (black) pixels and the actual pixel value in an image. This difference is then used to calculate the influence of the pixel on the output to understand its attribution to the output classification.

### 2.4.3 Method of Rationales

Another promising avenue of research in interpretability specifically in NLP, is the approach of rationales (Lei et al., 2016; Bastings et al., 2019). This method aims to understand how a neural network-based classification model reaches its prediction by identifying the words or phrases in a sentence or paragraph that are most essential for the model to achieve the correct sentence classification. Lei et al. (2016) define these rationales as being short and coherent pieces of text that alone must be sufficient for making the same prediction as the original text. In order to identify the tokens that comprise the rationale, the model must learn a binary mask over the original input sentence that learns to mask words which do not contribute to the rationale. Given the discrete nature of a token selection or masking task, these models must get around the non-differentiable nature of this task, however, Bastings et al. (2019), discussed in subsection 3.2.1, utilise a novel distribution, the Hard Kumaraswamy, to enable the differentiable binary masking of tokens. Given that the method for sentiment transfer proposed in this dissertation requires that the sentiment-specific words in a sentence be identified, the method of 'rationales' presents a promising avenue for finding these words.

Figure 2.5 gives a visual example of how the method of rationales might select a set of simplified rationales that "explain" the classification of a model. The difference between this method and Integrated Gradients, as well as DeepLift, is explained briefly in the following section.

**Positive**: the food here is **amazing** and the **service excellent!**

**Negative**: the waiter was **rude** and I was **left feeling very disappointed.**

Figure 2.5: An example of the words selected as the rationales in two negative and positive classified sentences, highlighted in **bold green**

### 2.4.4 Comparison of Methods

All of the proposed methods of interpretability reviewed in this dissertation comprise unsupervised methods of learning. This is due to these models being trained not on a primary task where there are "correct" interpretability annotations to compare against, but by these models being derived by utilising pretrained models (Sundararajan et al., 2017; Shrikumar et al., 2017) or by being trained at the same time as the underlying classification model (Lei et al., 2016; Bastings et al., 2019).

One of the challenges of utilising either the Integrated Gradients (Sundararajan et al., 2017) or DeepLift (Shrikumar et al., 2017) interpretability methods is that both methods provide a real number indicating the degree of impact a feature has on the output. In the case of language tokens, the continuous number assigned to each token is challenging to interpret in terms of which tokens have a noteworthy impact on the classification of the model. Thus, the researcher must decide on a cutoff level for whether an input can be seen to be significantly impacting the output of a model. In contrast, the method of rationales (Lei et al., 2016; Bastings et al., 2019) discretely chooses which tokens do and do not have an impact on the output. Although the percentage of tokens chosen is dependent on a set of hyperparameters set by the researcher, the model still learns to identify discrete tokens that lead to a classification. This method is therefore better suited to the goal of creating an automated pipeline for deleting different proportions of sentiment-specific words from sentences and was used in this dissertation.

## 2.5 Chapter Summary

This chapter has presented an overview of the relevant literature in the NLP domain for this dissertation, providing an introduction to neural network methods in NLP, pretraining and transfer learning in NLP, sentiment and style transfer with non-parallel corpora and finally, methods in the interpretability literature that are useful for the sentiment transfer challenge presented in Chapter

1. By highlighting relevant concepts in the literature, the foundation has been laid for the Methods chapter which follows.

# CHAPTER 3

# METHODS

This dissertation builds on previous work (Li et al., 2018; Sudhakar et al., 2019; Wu et al., 2019), employing a modified version of the DRG model (Li et al., 2018). The DRG method utilises a heuristic for removing the sentiment words from a sentence. The model then (re)generates a new sentence utilising a sentiment marker along with the content of the source sentence. This dissertation utilises a new approach for identifying the sentiment-specific words to remove, namely the method of rationales (Lei et al., 2016; Bastings et al., 2019). This dissertation also explores using a pretrained Transformers model, BART (Lewis et al., 2020), to transfer sentences from one sentiment to another. The model is first fine-tuned as a DAE (Vincent et al., 2008). It then generates a synthetic dataset which is used in the final stage of training to fine-tune the model in a self-supervised manner.

This chapter begins by presenting the datasets used in this dissertation. It then provides an in-depth discussion of the two methods used for identifying and removing the sentiment-specific words in the sentences, as well as the two models compared. After explaining the different methods used, it provides an overview of the full training pipeline employed in the research.

## 3.1 Data Sources

This dissertation utilises two datasets commonly used in the sentiment transfer literature. These datasets comprise a truncated version of Yelp and Amazon reviews that have been categorised as either positive or negative (He and McAuley, 2016). Concretely, where a review received a rating of four or five, it was classified as positive with a rating of one or two classified as negative. Reviews with ratings of three are excluded due to their being considered as ambivalent. On download these sentences are already pre-tokenized and, given that they are used in this pre-tokenized state in other research, they were left in this state for this dissertation. Table 3.1 shows the summary statistics for the datasets.

The data utilised is the same as the data from the original DRG paper (Li et al., 2018), thus the imbalance in positive and negative examples for the Yelp dataset was kept as is. The reference dataset for both Yelp and Amazon comprises a set of human annotated sentiment transfers along

| | Yelp | Amazon |
|---|---|---|
| Positive Sentences | 270,000 | 277,000 |
| Negative Sentences | 180,000 | 278,000 |
| Total Train Sentences | 450,000 | 555,000 |
| Dev Sentences | 63,000 | 2,000 |
| Test Sentences | 1,000 | 1,000 |
| Average Sentence Word Count | 7.9 | 13.8 |

Table 3.1: Summary statistics of the Yelp and Amazon reviews datasets used in this dissertation. Values for the number of sentences in each subset are rounded to the nearest 1,000.

| Yelp | |
|---|---|
| Sentiment | Sentence |
| Original Positive | it 's small yet they make you feel right at home. |
| Reference Negative | it's small yet they make you feel like a stranger. |
| Original Positive | i will be going back and enjoying this great place! |
| Reference Negative | i won't be going back and suffering at this terrible place! |
| Original Negative | but it probably sucks too! |
| Reference Positive | but it probably doesn't suck too! |
| Original Negative | the wine was very average and the food was even less. |
| Reference Positive | the wine was above average and the food was even better |
| Amazon | |
| Sentiment | Sentence |
| Original Positive | washing it is a breeze and just takes a second. |
| Reference Negative | washing is a pain and takes several hours. |
| Original Positive | i wore my first one out by pouring hot gravies in it. |
| Reference Negative | i wore my first one out by using it normally. |
| Original Negative | this is honestly the only case i ve thrown away in the garbage. |
| Reference Positive | this is honestly the only case i've kept for so long. |
| Original Negative | so i guess it s really a matter of preference. |
| Reference Positive | so I'm sure it s really a matter of preference. |

Table 3.2: Examples of original and human references in Yelp and Amazon datasets

with the original sentence. These human sentiment transfers were used to calculate the automatic evaluation BLEU scores for content consistency with the model-generated sentiment transfers. This is consistent with past research (Li et al., 2018; Wu et al., 2019; Sudhakar et al., 2019).

Table 3.2 provides a set of positive and negative examples from the datasets, as well as their reference human sentiment transfers. We notice that whilst it is relatively easy to grasp the sentiment of the Yelp sentences by reading them, this is not necessarily the case with the Amazon sentences. This challenge is explored in further detail throughout this dissertation.

## 3.2 Text noising

Text noising in this dissertation refers specifically to the removal of sentiment-specific words from the sentences used as input to the models. The term "noising" is used because both the baseline DRG and experimental BART models are trained as DAEs. For clarity, they are trained to take a sentence that has been noised - by either completely removing sentiment-specific words in the case of the baseline DRG model, or by replacing these words with the <mask> token in the case of the BART model - and to denoise that sentence by reconstructing it. For the DRG model, this is the entirety of its training, whereas for BART, this represents one stage in its training pipeline.

As noted in section 1.2, this research compares the results of two noising methods on the text before feeding the sentences as input to a neural model which acts as a DAE (Vincent et al., 2008). Whilst the first of the methods incorporates machine learning, the second method, as utilised by Li et al. (2018), is a heuristic-based approach. The first method employs the method of rationales (Bastings et al., 2019) to mask or delete a proportion of tokens in each sentence. This method learns a set of binary masks over the words in each sentence. The second approach, which is used as the baseline, employs a simple heuristic to identify positive and negative n-grams and to mask or delete these from the sentences (Li et al., 2018). This approach, called the saliency method in this dissertation, calculates a score for an n-gram based on how often it occurs in the positive and negative sentences. N-grams are then selected for removal depending on a saliency score cutoff. Both of these methods are explored in detail in this section.

### 3.2.1 Rationales Noising

For identification of sentiment tokens, this dissertation utilises the 'interpretable binary predictions' method of Bastings et al. (2019), which we simply refer to as the 'method of rationales' throughout this dissertation. This method builds on the initial rationales work of Lei et al. (2016), however, the method of Bastings et al. (2019) is more efficient and allows one to select the percentage of tokens, relative to the full sentence, that will be used in the rationale. This hyperparameter of the model is adjusted to create five subsets of data for each of the datasets used in the dissertation. Concretely, we use the model to noise the Yelp and Amazon datasets such that $15\%, 20\%, 30\%, 40\%$ and $50\%$ of the sentences are, on average, replaced with a <mask> token. In the case of the data

fed to the DRG model, these masked words are simply removed.

In implementing the method of rationales, both Lei et al. (2016) and Bastings et al. (2019) treat the distribution indicating whether a word is included in the rationale as a latent binary variable, $Z$. Here each $z_i$ takes on a value of either 0 or 1, indicating respectively whether a token is excluded or included in the rationale. Then, for each sentence, the words in the source text $x$ that do not form part of the rationale are masked out, with the unmasked words used as input to a trainable classifier. This can be formulated by the element-wise Hadamard product $\odot$ of the masks, $z$ with the source sentence tokens, $x$. The masks, which are represented by the latent variable $Z$, can be seen to be derived from a Bernoulli distribution (Bastings et al., 2019). The learning of the masks and therefore the rationales is defined as

$$Z_i|x \sim \text{Bern}(g_i(x;\phi)), \tag{3.1}$$

$$Y|x,z \sim \text{Cat}(f(x \odot z;\theta)), \tag{3.2}$$

where Bern() refers to a Bernoulli distribution and Cat() refers to a categorical distribution, $x$ refers to the original sentence, $z$ to the learned mask vector for each sentence, and $Y$ to the classification for the sentence. $g_i(x;\phi)$ and $f(x \odot z;\theta)$ indicate neural network models where both models are built with the LSTM (Hochreiter and Schmidhuber, 1997) architecture and are parameterised by $\phi$ and $\theta$ respectively.

The primary challenge with the original rationales model (Lei et al., 2016) is that to find the optimal rationale for a sentence would require marginalising over all of the rationales, which is intractable. In the original paper, Lei et al. (2016) approximate the gradient by sampling a number of rationales to get the expected gradient and using the REINFORCE algorithm (Williams, 1992) to get gradient estimates. However, this method can be plagued by issues of high variability due to high variance in the gradient estimates.

To overcome this issue, Bastings et al. (2019) employ a modified version of the Kumaraswamy distribution (Kumaraswamy, 1980) which the authors call the Hard Kumaraswamy distribution. This model is both continuous whilst also displaying discrete behaviour which enables it to create a mask for words within a sentence. This distribution is also able to utilise the reparameterization

trick (Kingma and Welling, 2014) during training. The reparameterization trick allows one to calculate the expected gradient over a stochastic element of a neural network by decomposing that stochastic element into a set of deterministic parts. Concretely, if we were to examine the case of a univariate Gaussian by decomposing the stochastic element into the sum of two learned deterministic variables with the second multiplied by a sampled value, we are able to differentiate with respect to samples drawn from this element. This is represented as

$$\begin{aligned} z \sim p(z|x) &= N(\mu, \sigma^2) \\ &= \mu + \sigma\epsilon \\ \text{where } \epsilon &\sim N(0,1), \end{aligned} \tag{3.3}$$

where we see that if $z$ comes from a normal distribution, we can reparameterise it as comprising a deterministic mean, $\mu$ and deterministic standard deviation, $\sigma$, with the latter multiplied by a random normal variable $\epsilon$. Thus, $z$ is still differentiable because $\mu$ and $\sigma$ are deterministic and thus differentiable. Using the reparameterization trick, the authors replace Equation 3.1 with a differentiable distribution,

$$Z_i|x \sim \text{HardKuma}(g_i(x;\phi)), \tag{3.4}$$

where $g_i(x;\phi)$ comprises two single layer feed forward neural networks over the hidden states from the bi-directional LSTM (Hochreiter and Schmidhuber, 1997), which processes the word embeddings of the original sentences. Here $g_i(x;\phi)$ learns two parameters, $a$ and $b$ that control the shape of the Hard Kumaraswamy distribution. This has probability density and cumulative density functions that are defined as

$$f_K(k;a,b) = abk^{a-1}(1-k^a)^{b-1} \tag{3.5}$$

$$F_K(k;a,b) = 1-(1-k^a)^b \tag{3.6}$$

$$F_K^{-1}(u;a,b) = (1-(1-u)^{\frac{1}{b}})^{\frac{1}{a}}. \tag{3.7}$$

Although the original Kuma distribution only takes on values within the range $(0,1)$, the authors generalise the support of the distribution and, due to the simple closed form expression of the CDF (Equation 3.6), are able to reparameterise the distribution. This is done by sampling from

$U(0,1)$ and applying the inverse transform in Equation 3.7 to sample from the distribution. The distribution is also shifted and scaled to allow the support to encompass 0 and 1 using two hyper-parameters, $l$ and $r$. To limit support to within the closed interval $[0,1]$, the authors then apply a hard sigmoid transformation to the Kuma distribution with the adjusted support, resulting in the Hard Kumaraswamy distribution.

The objective for the overall rationales-based model of Bastings et al. (2019) comprises two parts. First, the model aims to maximise the lower bound on the log-likelihood of the data, $\log P(y|x)$, which is the classification prediction $y$ conditioned on the sentence $x$. Using Jensen's Inequality, this is derived as

$$\begin{aligned} \log P(y|x) &= logE_{P(z|x,\phi)}[P(y|x,z,\theta)] \\ &\geq E_{P(z|x,\phi)}[\log P(y|x,z,\theta)] = \varepsilon(\phi,\theta). \end{aligned} \tag{3.8}$$

Second, the model aims to induce sparsity in the words being chosen so as to prevent the model from choosing the entire string of words for the rationale. The original implementation of the model also has an additional loss which aims to choose longer contiguous stretches of words over non-neighbouring words. This is based on the idea that rationales should comprise coherent pieces of text (Lei et al., 2016; Bastings et al., 2019). However, we removed this objective as we assumed that choosing single words, as opposed to choosing contiguous stretches of words, is more likely to lead the model to find sentiment-specific words in the text. The sparsity loss is defined as

$$\lambda \sum_{i=1}^{n} z_i, \tag{3.9}$$

where $\lambda$ is a hyper-parameter and each $z_i$ takes on either a value of 1 or 0. Thus the total loss optimised is

$$\min_{\phi,\theta} -\varepsilon(\phi,\theta) + \lambda \sum_{i=1}^{n} z_i. \tag{3.10}$$

The model learns to optimise $\min_{\phi,\theta} -\varepsilon(\phi,\theta)$ by training both the classifier and the binary masking model simultaneously. The words identified with this method are then masked out of, or deleted from, the source sentences and then fed to either the BART model or baseline DeleteOnly model

respectively.

### 3.2.2 Salience Noising for Baseline

In the saliency noising method (Li et al., 2018), the authors use an approach for locating and deleting sentiment-specific words which they refer to as finding the *salience* of the words or n-grams that they extract from sentences. This approach is conceptually similar to a TFIDF approach where words that have higher discriminative power are weighted more highly by the model. Here the salience of an n-gram is calculated as

$$s(u, v) = \frac{\text{count}(u, D_v) + \lambda}{(\sum_{v' \in \nu, v' \neq v} \text{count}(u, D_{v'})) + \lambda}, \tag{3.11}$$

where $D = (x_1, v_1), ..., (x_m, v_m)$ is the set of sentence-sentiment attribute pairs, $x_i$ is a sentence, $v \in \mathcal{V}$ are the attribute markers - or sentiment tokens - such that $\mathcal{V}$ = "positive", "negative", and $D_v$ is the corpus in one of the sentiments. Concretely, for each corpus comprising sentences of each sentiment, the occurrence of each n-gram of a defined size is counted in that corpus. The same n-gram is then counted in the corpus of the opposite sentiment. The salience ratio is thus the ratio of how many times an n-gram occurs in one of the corpora relative to the other, with high salience ratios being indicative of that n-gram having high discriminative power for that attribute. The smoothing factor $\lambda$ prevents numerical underflow if the n-gram does not occur in the corpus in the denominator. Based on the results from the original paper (Li et al., 2018), we consider spans of up to 4 words for the length of the n-grams for the saliency-based noising procedure and set the attribute marker threshold to 15 and 5.5 for Yelp and Amazon respectively.

## 3.3 Sentiment Transfer Models

### 3.3.1 BART model

This dissertation utilises a pretrained BART model (Lewis et al., 2020) to test the potential improvement with using a pretrained language model over training a language model from scratch for sentiment transfer. This model can be seen as a generalisation of BERT, with the encoder employing bidirectional attention combined with GPT as the decoder, which utilises masked attention

and auto-regressive word prediction for training. BART is trained with a DAE objective similar to BERT, however, unlike BERT, BART is also trained auto-regressively on a GPT-like next word prediction task. Different from the original Transformer architecture, each layer of the decoder performs cross-attention over only the final hidden layer of the encoder - as opposed to each intermittent layer - and BERT utilises an additional feed-forward layer prior to word prediction, which BART does not (Lewis et al., 2020). In spite of BART comprising both an encoder and decoder, it only uses an additional 10% of parameters when compared with BERT (Lewis et al., 2020).

Concretely, noised sentences are fed into the BART encoder and the decoder must learn to predict the next word in the passage. The sentences are noised by replacing words with a <mask> token as well as with word permutation, however, we only masked the words without conducting word permutation in this dissertation. This latter step did not seem necessary as the model had already learned to generate fluent sentences during pretraining. The original BART model is trained utilising the same loss as GPT, minimising the negative log likelihood of the original document $x$, however, some of the tokens $x_i$ are replaced by the <mask> token. As we are conditionally generating text, we prepend a sentiment token to the decoder in the model. Thus, the model is fine-tuned to minimise the negative log likelihood when predicting the next word in masked sentence $x$, conditioned on the sentiment token $s$. This loss is simplified as

$$L = -\sum_{(x,S)\in D} \log p(x|m(x,s),s), \tag{3.12}$$

where $(x, S) \in D$ represents the dataset of sentences $x$ with sentiments $S$, $m(x, s)$ represents the original sentence with sentiment words masked out, $x$ the sentence to be reconstructed and $s$ the relevant sentiment token.

This model was chosen for this dissertation for two key reasons:

1. A noteworthy proportion of previous research in sentiment and style transfer has been built off autoencoder objectives (Fu et al., 2018; Logeswaran et al., 2018; Wang et al., 2019) with a subset of these utilising a DAE architecture (Subramanian et al., 2018; Li et al., 2018; Wu et al., 2019). Thus, the BART model, positioned as a DAE by the paper's authors, presented itself as a noteworthy potential architecture for experimentation in this domain.

2. As a generative, sequence-to-sequence model (Sutskever et al., 2014), BART appears well-designed for the task of generating fluent sentences in a new sentiment, acting as a sentiment transfer model.

The details of how the model was trained are explored further in the section 3.4.

### 3.3.2 Baseline Model: Delete Retrieve Generate

For comparison with past work, this dissertation utilises the DeleteOnly version of the DRG model (Li et al., 2018). Of the versions of the model utilised in the DRG paper the DeleteOnly version is most suitable for comparison as it learns to generate a sentence with a different sentiment conditioned only on the noised sentence and a sentiment token. In this way, the model acts as a DAE similar to the BART model.

This model is trained by first processing the input data. This is done by removing the sentiment words associated with the source sentiment of a sentence to create a new sentence comprising only the content words. The model then learns to reconstruct the original sentence using only the content and sentiment token of the original. Here the loss is defined as

$$L = -\sum_{(x,s)\in D} \log p(x|c(x,s),s), \tag{3.13}$$

where the sentence content is expressed as $c(x, s)$, $x$ is the original sentence and $s$ is the attribute marker - or sentiment token - of the source sentence. For clarity, $c(x, s)$ refers to the noised version of the sentence. The model has been constructed as an LSTM with a single bidirectional layer. As noted in Equation 3.13, the model learns to minimise the negative log-likelihood of the reconstruction of the sentence, given the content of the source sentence along with its relevant attribute or sentiment marker. Here the model acts as a DAE on the noised sentence. Concretely, the encoder first encodes a representation of the source content. This final hidden layer of the encoder is then concatenated with a learned attribute embedding and passed to the decoder. Finally, the decoder learns to reconstruct the original sentence based on the learned representation for the content and sentiment marker of the source sentence. By learning to reconstruct the original sentence conditioned on only the source content and attribute marker, the model aims to learn

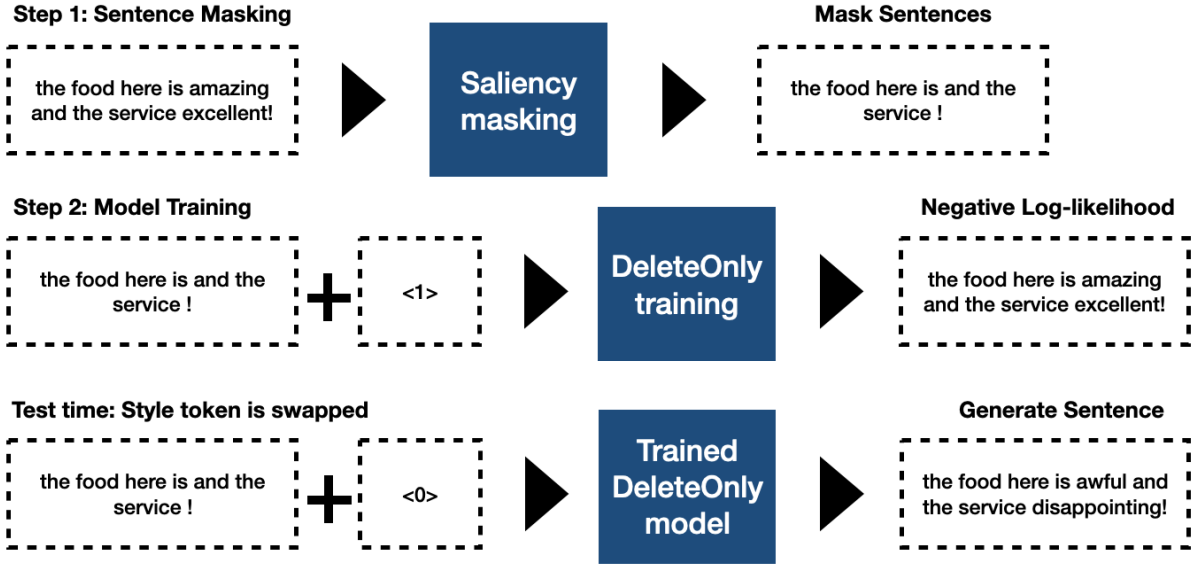


Figure 3.1: The DeleteOnly model training process including saliency noising

both how to construct grammatically correct sentences, as well as to construct sentences guided by a sentiment token. Although the LSTM model is trained from scratch, the model utilises a set of pretrained GLoVE embeddings (Pennington et al., 2014) for the token embeddings. The training process is shown in Figure 3.1. Note here that model uses 1 and 0 for the positive and negative sentiment tokens respectively.

The proposed BART and baseline models are similar in that they both aim to minimise the negative log likelihood of the predicted distribution of tokens in a sentence conditioned on previous tokens and a sentiment attribute marker. Similar to BART, the sentiment tokens are prepended to the beginning of the sentence at the decoder section of the DeleteOnly model architecture.

There are, however, three primary differences between the BART model and the baseline model. Firstly, the BART model employs a pretrained Transformer (Vaswani et al., 2017) architecture whilst the DeleteOnly model utilises an LSTM trained from scratch, albeit using pretrained word embeddings. Secondly, sentiment words are replaced by the <mask> token in the input data for the BART model whilst these words are simply removed from the input data for the DeleteOnly model. Finally, the BART model has an extra training stage where it generates sentences in the opposite sentiment and uses these as a self-supervised training set for sequence-to-sequence learning. This

is explored further in the following section on the training pipeline.

## 3.4 Training Pipeline

This section provides an overview of how the stages in the training pipeline are connected for the BART model. The pipeline for the baseline comparison model, shown in Figure 3.1, is relatively simple and is not discussed in detail here. The training pipeline for the BART model comprises five steps, namely: training the classifier, training and using the rationales sentence noiser, training the BART DAE, generating a quality parallel corpus, and finally training the BART self-supervised sentiment transfer model.

### 3.4.1 Classifier Training

The first stage of the training pipeline involves training the BART-encoder-based classifier to classify the sentiment of the sentences in the corpus. Un-noised, original sentences are fed to the classifier. Given that the sentences in both datasets are already labelled, this is a relatively trivial task. This step is shown as step 1 in Figure 3.2.

### 3.4.2 Sentence Masking

After training the classifier, the rationales model of Bastings et al. (2019) is trained to learn which words or tokens comprise the rationales for a classifier to make its predictions. Here, a separate

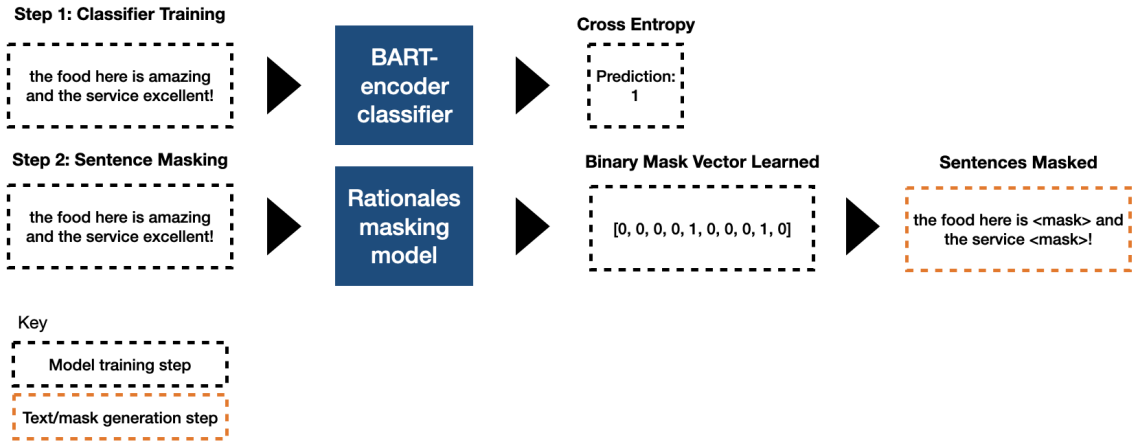


Figure 3.2: Overview of classifier pretraining and sentence noising processes in the BART training pipeline

classifier to the BART-encoder-based classifier is trained simultaneously with the rationales model as outlined above. The output of this step in the pipeline comprises a vector of masks for each sentence, where 1 indicates that word forms part of the rationale for a sentence and 0 indicates that the word does not. Every token in a sentence at the position of a 1 in the mask vector was replaced with <mask> token. This process is shown in step 2 of Figure 3.2

A separate noising process was conducted for the saliency-based noising. This allowed us to compare the efficacy of the rationales noising versus the saliency noising as per research question 1. The details of this approach to noising are given in subsection 3.2.2.

### 3.4.3 DAE Model Training

The first step in the BART training pipeline trains the model as a DAE to reconstruct the source sentences with <mask> tokens in the places of the words identified as sentiment words. The tokens in the source sentence are masked and the model is tasked with reconstructing the sentence and filling in these masked tokens. Here the first token as input to the decoder represents the sentiment of the source sentence input to the encoder, namely either <pos> or <neg> for positive and negative respectively. The model is trained to optimise the negative log-likelihood of the source sentence that it aims to reconstruct.

Although the model optimises the loss described in Equation 3.12, training is controlled by an early stopping criterion. This criterion is based on the accuracy of the generated validation sentences as measured by the pretrained classifier. This early stopping criterion is based on the accuracy of the *transferred* sentences, where the model stops training when accuracy does not improve for a number of consecutive validation steps. Concretely, after a set number of batches the reconstruction-focused language model runs a validation test on a held out development set. From this set, 1,000 masked sentences are sampled at random and are fed to the partially fine-tuned model. The model then transfers these sentences from one sentiment to another by seeding the decoder with the opposite sentiment token of the source sentence. The generated (transferred) sentences are subsequently fed to the classifier trained in step 1. This process of sentiment transfer for the DAE model is shown in Figure 3.3 and step 3 in Figure 3.4 provides a high level overview of this process.

The model also outputs a set of reconstructed sentences at each validation step to ensure that

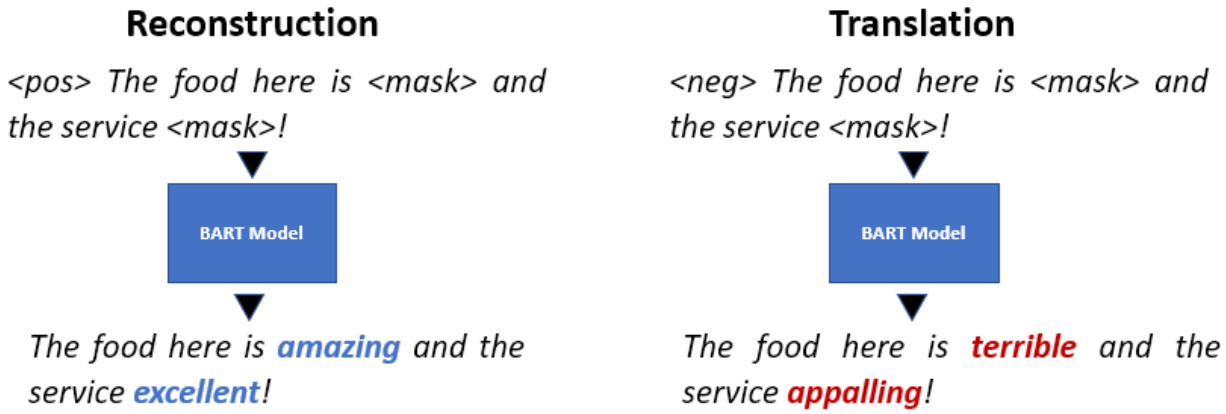


Figure 3.3: An example of the sentiment transfer and reconstruction process during BART DAE training

training was progressing as expected. The sole purpose of these outputs is to enable manual inspection of the training process.

### 3.4.4 Parallel Corpus Generation

Once the early stopping criterion of the DAE model training has been met, this training stops. The fine-tuned DAE model then takes as input all of the noised sentences from the training set and generates a new training set of matched source-sentiment, target-sentiment sentences. To create a matched training set of quality examples, the model only keeps the generated sentences that have been classified as having been accurately transferred into the target sentiment, based on the results of the pretrained classifier. A parallel corpus is generated here to improve the results of the final model by enabling an additional sequence-to-sequence training step. This is shown in step 4 in

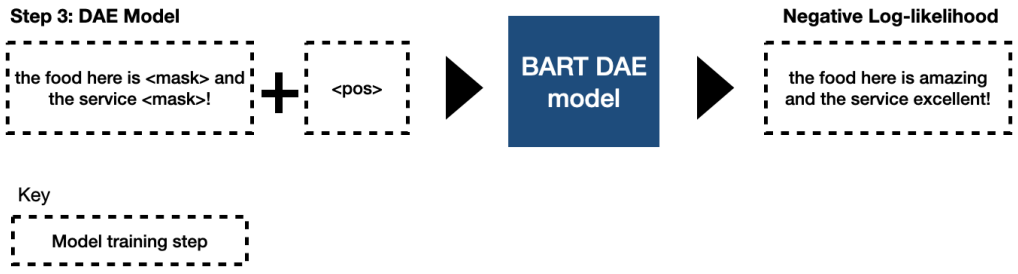


Figure 3.4: Overview of DAE training in the BART training pipeline

Figure 3.5.

### 3.4.5 Self-Supervised Training

At the final stage of the training pipeline, the BART model learns to transfer the sentences from one sentiment to another in a self-supervised fashion. Here, we use the term self-supervised as the parallel training set was created entirely by the model and did not exist prior to training. For clarity, as the model that creates the supervised dataset is trained in an unsupervised fashion and the final model in a supervised fashion, this is a self-supervised approach.

The self-supervised model is instantiated with the weights of the best performing model from the previous DAE reconstruction step in the pipeline. Note here that 'best performing' pertains to performance as measured by sentiment transfer accuracy on the validation set, where the best performing weights are kept subject to the early-stopping criterion. This model also utilises un-noised sentences as the source sentences fed to the encoder. This was done because the model achieved better sentiment transfer results on the sequence-to-sequence sentiment transfer when un-noised sentences were used in the input.

Here the model learns to transfer from one sentiment to another with parallel sentences in each sentiment. This loss is defined as

$$L = -\sum_{(x,y,s,s')\in D'} \log P(y|x,s'), \tag{3.14}$$

where $(x, y, s, s') \in D'$ represents the new dataset of generated parallel sentences. $x$ is the source sentence with sentiment $s$ and $y$ is the transferred sentence with sentiment $s'$ where $s \neq s'$

Figure 3.5 gives an outline of the training procedure for the self-supervised part of the training pipeline. The final training step of the model is the same process that is followed at test time.

### 3.4.6 A Note on Validation Sentence Generation

The generation of sentences as part of the model training and validation process, and as a means of generating a novel parallel corpus, is an essential sub-process in the pipeline as noted in subsections 3.4.3 and 3.4.4. During these sub-processes, the partially-trained model was set to evaluation mode

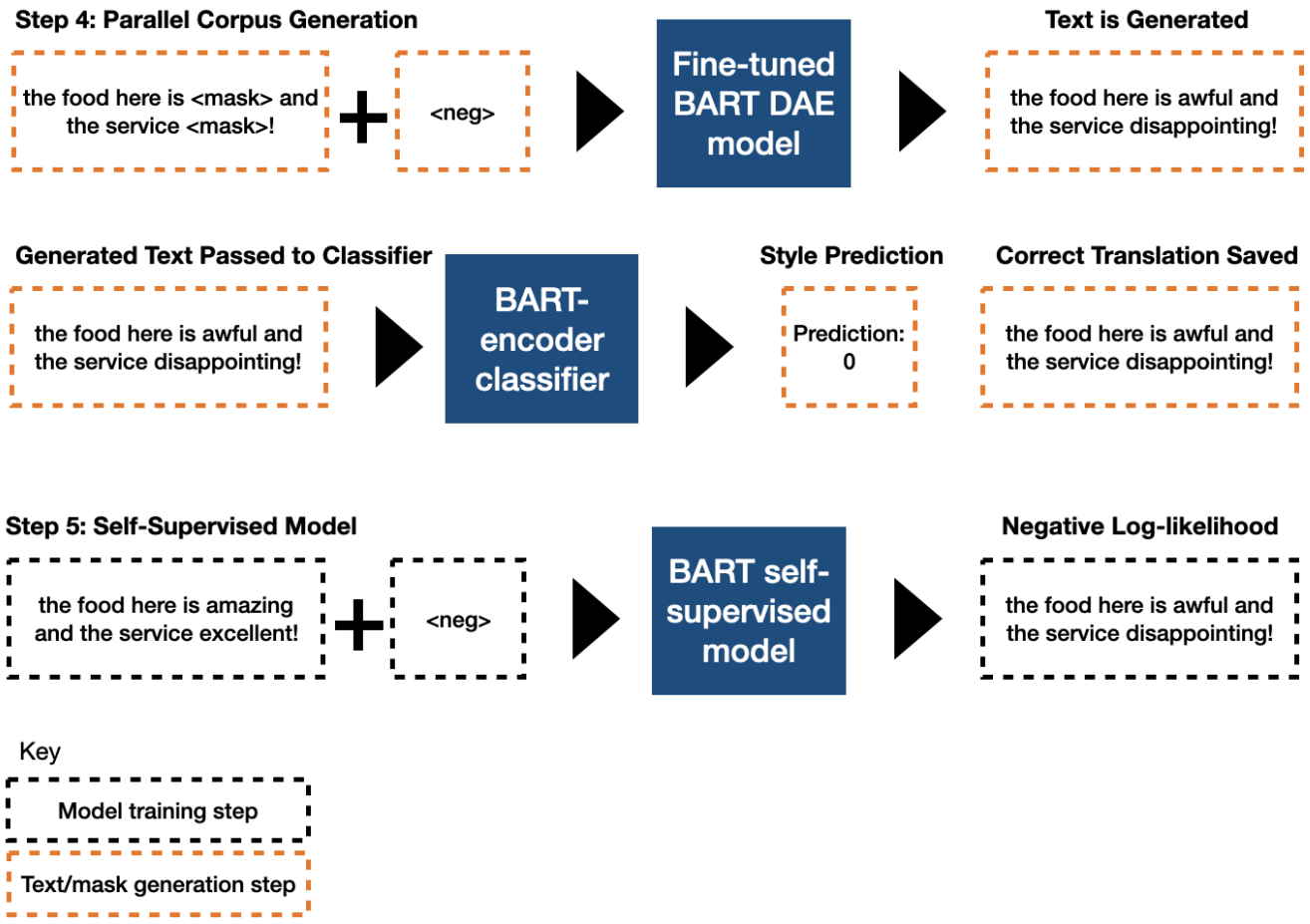


Figure 3.5: Overview of self-supervised training in the BART training pipeline. At test time the model uses step 4 to generate test sentences, except sentences are not masked.

and sentences were generated auto-regressively. This was done by passing an appropriately noised or un-noised sentence into the encoder, seeding the decoder with the appropriate sentiment token and then generating a set of sentences one token at a time using greedy decoding. Numerous next token selection methods were used, however, greedy decoding returned the best results. Sentences were generated in batches of 64, with minor formatting performed on the sentences where double spaces and some excess punctuation were removed. As noted previously, during the validation and generation process, intermittent results were saved so as to qualitatively examine the progress of the model as it learned.

### 3.4.7 A Note on Model Training Paradigms

Not including reinforcement learning, model training falls within four learning paradigms, namely unsupervised, semi-supervised, self-supervised and supervised learning. Unsupervised learning refers to learning patterns in a dataset when no labels, response variables or annotations are present. Supervised learning refers to learning patterns in a dataset when labels, response variables or annotations are present (James et al., 2013). The step used to identify and mask or delete sentiment words from sentences is referred to as unsupervised in the literature Li et al. (2018). This follows logically given that no annotations are used that indicate which words comprise the sentiment words in the sentiment word identification tasks. In general a DAE is considered an unsupervised learning task (Vincent et al., 2008) and in the context of sentiment and style transfer using a DAE is described as unsupervised Li et al. (2018).

The combination of an unsupervised task and supervised task is often considered in the literature as a semi-supervised task (Radford et al., 2018). Within NLP, the use of this term is prevalent when researchers describe an unsupervised pretraining objective followed by a supervised fine-tuning step. In spite of this combination of unsupervised and supervised training, the final step of the training process for the BART model in this dissertation is referred to as self-supervised for two reasons. Firstly, the process is supervised given that the model is presented with an input and the desired output we wish the model to map the input to. However, it is considered self-supervised because the model has generated the dataset that it trains on in this final step entirely by itself after the initial unsupervised training. If the model had not generated its own dataset, then this could be considered a semi-supervised task.

## 3.5 Chapter Summary

This chapter has presented a discussion of the datasets, the data noising methods, as well as the sentiment transfer models used in this dissertation. The technical details of each of these approaches was presented. The chapter ended with a discussion of the various steps in the BART training pipeline. This chapter has laid the foundations for the following chapter, which outlines how each of these methods is set up for training before presenting the BART model training curves.

# CHAPTER 4

# EXPERIMENTAL SETUP

This chapter presents an overview of the experimental setup for the models, including hyperparameters used. This is followed by a discussion of the noising results of the rationales (Bastings et al., 2019) and saliency (Li et al., 2018) noising methods used, showing that there is some variance in the proportion of sentiment-specific words identified in each sentence by both the rationales and saliency noising methods. After discussing these two approaches for sentiment word identification, we then discuss the setup and training for the two models used for comparison in this dissertation. This chapter also presents a thorough discussion of the automatic and human evaluations used to assess the final outputs created in this dissertation. Finally, the chapter ends with a presentation of the BART model training curves. This provides insight into how the model learns for both the Yelp and Amazon datasets at the DAE and self-supervised parts of the training pipeline.

For clarity, we use a term such as *Yelp 40* to indicate the Yelp dataset that has been processed with the rationales model aiming to mask or remove 40% of the tokens in each sentence. *BART Yelp 40* on the other hand refers to the BART model trained with this noised version of the Yelp data.

## 4.1 Resources

### 4.1.1 Coding Approach

The code for this dissertation was written in Python, utilising the PyTorch framework and the HuggingFace (Wolf et al., 2020) open source pretrained language models, in particular the BART model of Lewis et al. (2020). The pretrained tokenizer and weights for the models are openly available for download, with the pretrained *bart-base* tokenizer and model weights used.

### 4.1.2 Training Resources

Initially experimentation was conducted utilising Jupyter Notebooks in Google Colabatory to make and test modifications to the BART architecture that were necessary for the research. Utilising this platform was vital for gaining access to a graphics processing unit (GPU), without which model

training would have been infeasible due to the slow pace of model learning on a CPU. This is due to the large nature of the underlying Transformer models and the significant speed-up enabled by the efficient matrix multiplication methods designed into a GPU.

Once initial experimentation had been completed, the bulk of the research was conducted on the Council for Scientific and Industrial Research (CSIR) high performance computing cluster, the CHPC. This provided us with access to a GPU and the ability to schedule in computing jobs. Research took approximately six months to complete. All models were trained on the Lengau cluster, utilising a virtual machine comprising a single GPU and 10 virtual CPUs.

## 4.2 Model Details

### 4.2.1 Rationales Noising

The Python code for the rationales model of Bastings et al. (2019) is used for identifying the sentiment-specific words to mask or delete for the Yelp and Amazon datasets. As noted in section 3.2, this process is referred to as noising. This code was obtained from the paper's associated GitHub page[1]. Some minor adjustments were made to the SST version of the code such that the model would only predict two labels instead of five. As noted in the Methods chapter, the loss that aims to ensure that the rationales are close to each other in proximity - and thus form coherent phrases - in the source sentence is removed from the training process. This is to ensure that the model focuses on finding sentiment-specific words that will be masked or deleted from sentences before they are input into the models. A batch size of 128 is used and the model is trained with an Adam optimizer with learning rate set to $2 \cdot 10^{-4}$ and number of epochs of training set to 20. Pretrained GLoVE embeddings (Pennington et al., 2014) are used, with the embedding size set to 300 as is the default in the original paper (Bastings et al., 2019). To test the impact of different levels of noising on sentiment transfer accuracy and content consistency of the generated sentences, five levels of rationales-based noising are used, namely 15%, 20%, 30%, 40% and 50%. These percentages represent the proportion of tokens that were replaced with the <mask> token in each sentence for the BART model or deleted for the DRG model. Each of these versions of the data are referred to as Yelp 15 through 50 and likewise Amazon 15 through 50. The models

[1] https://github.com/bastings/interpretable_predictions

trained from these different noising schemes of the data are likewise referred to using the naming convention BART Yelp 15, DeleteOnly Amazon 20 etc. As noted by Bastings et al. (2019), one of the advantages of their method in relation to the original rationales method of Lei et al. (2016), is that it allows the researcher to specify the expected proportion of tokens which are selected by the model in each sentence. This proportion is an additional hyperparameter specified by the researcher prior to training. Along with the five levels of rationales noising used to compare the models, an additional baseline noised version of the data is used that utilises the saliency method of Li et al. (2018). This is outlined further in the following subsection.

After learning the positions of the binary masks in the data, the selected words are replaced with the <mask> token, or in the case of the DRG baseline, simply deleted. Although the original BART model (Lewis et al., 2020) replaced random length spans of words with the <mask> token, this implementation masks a single word with a single <mask> token. This is chosen primarily to uphold the intuition that the sentiment of sentence can be localised to a few individual words. Table 4.1 shows an example of a number of original and subsequently masked sentences from both the Yelp and Amazon datasets.

Although Table 4.1 shows sentences noised utilising different levels of noising, it is apparent that the model does not always select exactly the percentage of tokens as expected. This is explored further in the following subsection.

### 4.2.2 Saliency Noising

The saliency noising utilises the PyTorch implementation[2] of the DRG model of Li et al. (2018). As noted in the Methods chapter, the saliency of each n-gram is calculated as

$$s(u, v) = \frac{\text{count}(u, D_v) + \lambda}{(\sum_{v' \in \nu, v' \neq v} \text{count}(u, D_{v'})) + \lambda}, \tag{4.1}$$

where this equation utilises the following hyperparameters: $\lambda$, the smoothing parameter, is set to 1; spans of up to four words are considered as attribute markers; and $\gamma$, the threshold for the saliency score, is set to 15 and 5.5 for Yelp and Amazon respectively. These settings are aligned with those from the original paper.

[2] https://github.com/rpryzant/delete_retrieve_generate

The saliency noising is conducted over both datasets where for each sentence in the dataset the most salient n-gram is removed, followed by the second most salient n-gram, and then the third until no more salient n-grams that scored above the threshold level of $\gamma$ are found in the source sentence. For the input data to the BART model, each word in each of the salient n-grams in the source sentence is replaced by a <mask> token. For the input data for the DeleteOnly model, these words are simply removed from the sentence.

### 4.2.3 Analysis of Noising Distribution

To understand the impact of the different levels of noising on the data, it is valuable to examine a histogram of the proportion of tokens in a sentence that are replaced by the <mask> token. The rationales model of Bastings et al. (2019) learns to optimise the accuracy of sentence classification whilst obeying the constraint of a user-defined expectation - or average - for the proportion of masked tokens in a sentence. This expectation is simply the number of tokens masked in a sentence,

| **Noised Data** | **Original** | **Noised** |
|---|---|---|
| Yelp 15 | wake up or you are going to lose you business. | wake up or you are going to <mask> you business. |
| Yelp 20 | very disappointed in the customer service. | very <mask> in the customer service. |
| Yelp 30 | very rude and once they get paid have no ethics. | very <mask> and once they get paid have <mask> ethics. |
| Yelp 40 | the bagels are fluffy and delicious. | the bagels <mask> <mask> and <mask>. |
| Yelp 50 | loved my brows and now i am ready to rock my feet for spring. | <mask> my brows and now i <mask> <mask> to <mask> <mask> <mask> for <mask>. |
| Amazon 15 | i cook every day and need a sturdy utensil. | i cook every day and need a <mask> utensil. |
| Amazon 20 | it does not add any bulk to the case. | it does <mask> add any bulk to the <mask>. |
| Amazon 30 | when i was incarcerated in a mexican prison, this was the only drink in the joint. | when i was incarcerated in a mexican <mask>, this <mask> the <mask> <mask> in the <mask>. |
| Amazon 40 | i vomit when i think of this product. | i <mask> when i think of this <mask>. |
| Amazon 50 | i purchased two and am glad i did. | i purchased <mask> and <mask> <mask> i <mask>. |

Table 4.1: Original and masked sentences with different levels of noising for the Yelp and Amazon datasets

| Noised Data | Yelp Mean | Yelp Std. Dev |
|---|---|---|
| Yelp 15 | 16.9% | 8.9% |
| Yelp 20 | 21.2% | 11.1% |
| Yelp 30 | 30.4% | 13.6% |
| Yelp 40 | 35.8% | 15.1% |
| Yelp 50 | 49.7% | 17.5% |
| Yelp Saliency | 32.4% | 24.3% |

Table 4.2: Mean and standard deviation of the proportion of tokens selected in the noised Yelp dataset

divided by the total number of tokens in the sentence. Figure 4.1 shows the distribution of the masked tokens for the Yelp dataset under different regimes of noising. What is clear here is that in spite of the rationales model aiming to meet a word selection constraint, there appears to be a noteworthy level of variance in the percentage of tokens selected per sentence, as well as some deviation from the stipulated word selection percentage. We also notice that the 40% and 50% noised versions of the datasets appear to have a bi-modal distribution, whilst the saliency-noised data is more uniform below about 70% noising, with a peak at the lowest level of noising.

Table 4.2 also highlights the different average levels of word selection and the associated standard deviation of the proportion of tokens selected across sentences for different noising regimes for the Yelp dataset. Unsurprisingly, the mean proportion of tokens selected increases as the user-defined constraint on the proportion of words selection to be achieved is increased. The standard deviation of the proportion of tokens selected also rises as the mean proportion of tokens selected increases. Recall that the proportion of tokens selected represents the proportion of *unmasked tokens* fed to the classifier during the rationales training. This is because we flip the binary mask from the rationales model to mask out or delete sentiment words and preserve content words to feed into the downstream sentiment transfer models.

In initial experiments we saw that occasionally the rationales model would collapse for the Yelp 15 version of the dataset, such that it would get stuck in a local minimum during training where it would learn to mask all of the words in corpus. It is surmised that this level of noising may be unstable for the Yelp dataset, perhaps being too low given the short length of the sentences at an average of 7.9 tokens.

The histograms for the Amazon data in Figure 4.2 show distributions that appear more normal

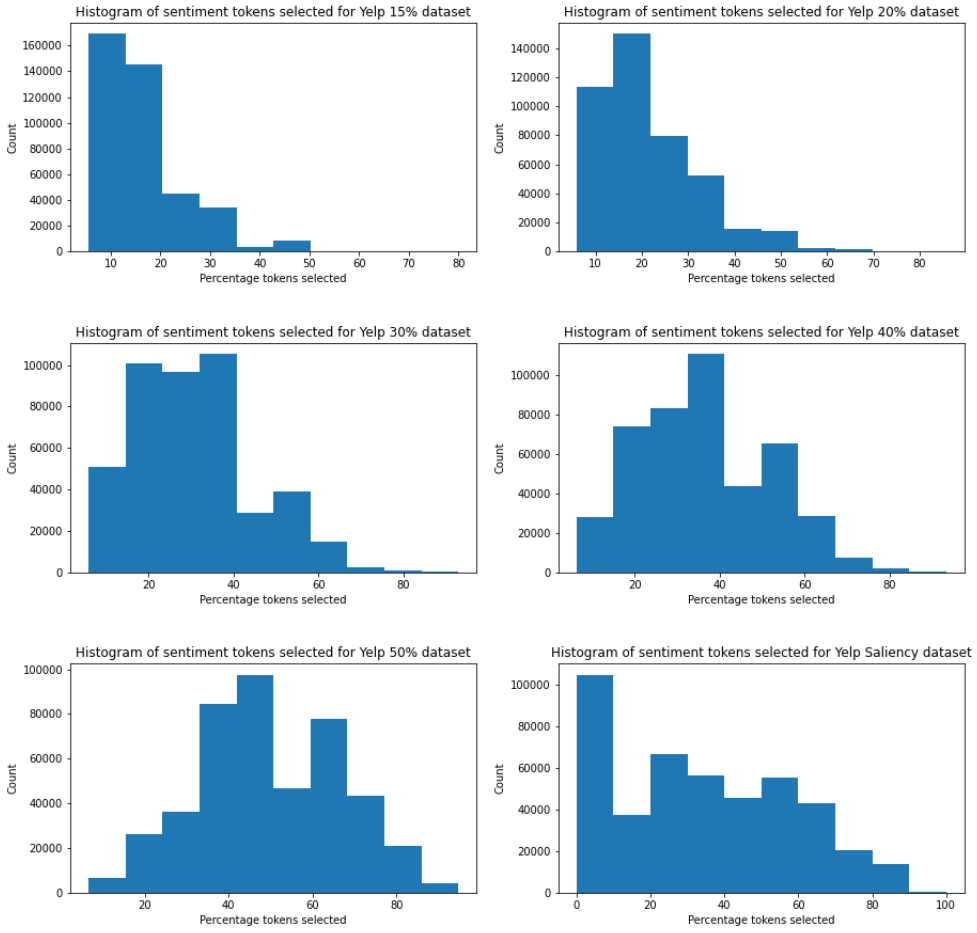


Figure 4.1: Histogram of the different proportions of tokens selected per sentence by noising scheme for the Yelp dataset

(Gaussian) around their expected means than did the Yelp histograms shown in Figure 4.1.

Table 4.3 highlights the different average levels of word selection and the associated standard deviation of the word selection across sentences for different noising regimes for the Amazon dataset. Here we see results that are in line with the user-defined levels of noising required by the model, similar to the Yelp results. Surprisingly, the standard deviation of the word selection dips for Amazon 50, not rising monotonically for the successively increasing mean word selection proportions,

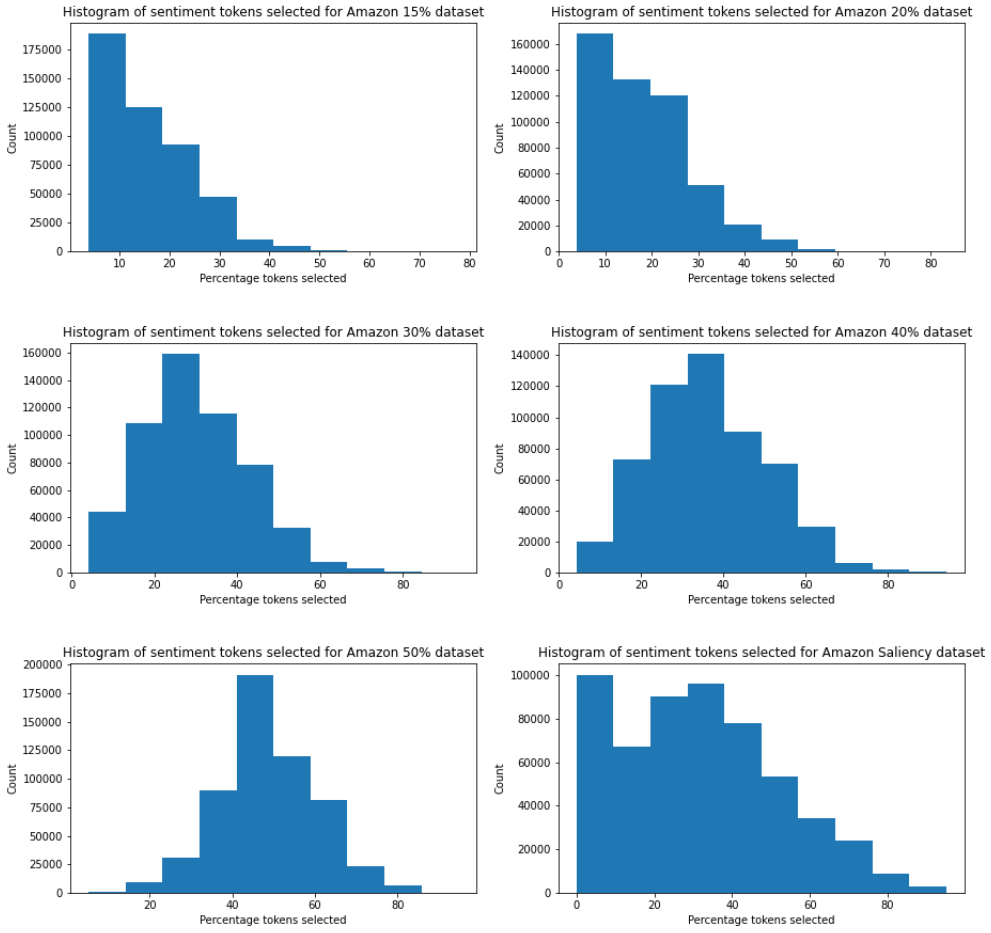


Figure 4.2: Histogram of the different proportions of tokens selected per sentence by noising scheme for the Amazon dataset

as happens with Yelp.

| Noised Data | Amazon Mean | Amazon Std. Dev |
|---|---|---|
| Amazon 15 | 16.0% | 8.8% |
| Amazon 20 | 18.3% | 9.9% |
| Amazon 30 | 29.9% | 12.3% |
| Amazon 40 | 36.4% | 13.7% |
| Amazon 50 | 49.3% | 11.9% |
| Amazon Saliency | 31.1% | 21.2% |

Table 4.3: Mean and standard deviation of the proportion of tokens selected in the noised Amazon dataset

#### 4.2.4 BART Classifier Training

The BART classifier is trained by extracting the encoder of a pretrained HuggingFace (Wolf et al., 2020) implementation of the BART model and training two fully connected feed forward layers on top of the model. Although some papers use a different architecture for text classification during training and evaluation (e.g. Sudhakar et al. (2019)), this design decision was taken early on in the research process to simplify the sets of architectures used. The tokenizer of the model was also pretrained, utilising byte pair encoding (Sennrich et al., 2016) to tokenize the inputs. The weights of the pretrained encoder are frozen whilst the weights of the two feed-forward layers are trainable. The model is trained utilising an Adam optimizer with learning rate set to $2 \cdot 10^{-5}$. The classifier is trained for two epochs, due to the accuracy on the validation set plateauing during the second epoch. This model achieved an accuracy of

1. 98.0% for the Yelp dataset
2. 88.4% for the Amazon dataset

As the classifier is trained to predict the sentiment of the original, fully-constructed sentences, the same classifier is used to make predictions on each of the outputs of the different models for each of the levels of noising.

#### 4.2.5 BART Sentiment Transfer Training

The Hugging Face (Wolf et al., 2020) pretrained implementation of the BART-base model is used. Given the generative nature of the dissertation, the pretrained BART For Conditional Generation version of the model is used. This comprises a language modelling head on top of the model and

consists of six encoder and decoder blocks in the encoder and decoder. There are 139 million parameters in the model. The dimensionality of the model is $d_{model} = 768$. The weights are optimized using the Adam optimizer with a learning rate of $2 \cdot 10^{-5}$. After initially running the model for three epochs during early experimentation for each step of the training pipeline, we chose to use early stopping to prevent the model from over-fitting. Based on empirical examination of the results, the early stopping criterion is set such that training terminated when the accuracy of the sentiment transfer output did not improve for three consecutive batches. Here accuracy is measured on a set of $1,000$ sentences transferred from one sentiment to another generated from a held-out development set of data. Validation is conducted every 250 batches. Although larger batch sizes were tried, memory constraints led to a batch size of 64 for training. Additionally, during training all of the weights of the model, as well as the token embeddings, are allowed to update. Given that two new tokens are added to the data, namely <pos> and <neg>, it is considered necessary for the model to learn a representation for these new tokens. The positional embeddings, however, are not trained as it did not appear necessary.

Due to a degree of stochasticity in the BART model results, each version of the model (e.g. BART Yelp40, BART Yelp50, BART Amazon 20 etc.) was trained three times and the final accuracy and BLEU results of the three runs averaged for reporting.

As noted in subsection 3.4.3 in the Methods chapter, the BART model is first trained as a DAE to reconstruct the noised sentences. Once the early stopping criteria is met on the sentences transferred from one sentiment to another during this training step, the model then generates a self-supervised parallel corpus of matching sentences in the positive and negative sentiment. This parallel corpus is then fed as input into the final stage of training. Here the fine-tuned model from the DAE phase is explicitly trained to transfer from one sentiment to another with the matched corpus. For clarity, we use the fine-tuned DAE model as the starting model for the self-supervised sentiment transfer training. This was done as it achieved better results than when fine-tuning the self-supervised model directly from the pretrained BART-base. As with the DAE phase, validation sentences are generated every 250 batches during the self-supervised training, with the same early stopping criteria as in the DAE phase.

During validation and testing, all sentences are generated using greedy decoding. This is done

based on empirical results where greedy decoding received preferable results on the final outcome. An example of the generated sentences is shown in section 5.3.

### 4.2.6 DeleteOnly Training

The DeleteOnly sequence-to-sequence model is trained utilising the PyTorch implementation (Pryzant et al., 2020) of the Bastings et al. (2019) paper. As per the BART model, this model is trained with various levels of noising from the rationales model (Bastings et al., 2019) as well as with data noised using the saliency method from the original DRG paper (Li et al., 2018). Unlike the word-piece tokenization of the BART model, tokenization is implemented by splitting full words on spaces, choosing a vocabulary size, and then utilising the top $k$ occurring words. Here $k$ is chosen as $16,000$, however, for both Yelp and Amazon less than $16,000$ unique tokens were found. Both of the original datasets had been partially tokenized, including having words such as *don't, i'll, can't* tokenized to *do nt, i ll, ca nt*. This pre-tokenization is maintained in the text for both the BART and DeleteOnly models.

The DeleteOnly model is trained utilising an Adam optimizer with learning rate of $2 \cdot 10^{-4}$ as per the default settings in the implementation. The model is trained for 70 epochs for both datasets. The embedding dimension is kept as the default of 128 and hidden dimension of the LSTM encoder and decoder is kept to 512. The final output of the model is assessed by passing a noised version of the test dataset through the model and having the model generate new sentences conditioned on the content words, as well as the desired sentiment token. In each case the sentiment token that the sentence generation is conditioned on is the opposite of the sentiment of the source sentence. These output sentences are then passed to the pretrained BART-encoder classifier to calculate the model sentiment transfer accuracy. An example of the final sentences in shown in Table 5.3.

## 4.3 Evaluation

### 4.3.1 Quantitative Evaluation

The outputs of the model are assessed based on the accuracy of sentiment transfer from one sentiment to another. As noted previously, we have used a pretrained and fine-tuned BART-encoder-based classifier to measure the accuracy of the sentiment transfer. Here, sentences are generated

by the fine-tuned BART model at the end of the training pipeline and these sentences are used as input to the classifier. The accuracy of the classifier is simply the number of sentences that are correctly transferred from one sentiment to the other over the total number of sentences in the test corpus.

Along with comparing the accuracy of the sentences transferred from one sentiment to another, it is also necessary to compare the degree of content preservation between the source and target sentences. For these purposes, we utilise the BLEU score (Papineni et al., 2002) when comparing the model-generated sentences with a set of gold-standard sentences transferred from one sentiment to another. BLEU is an acronym for bilingual evaluation understudy and has traditionally been used in machine translation tasks to identify how close to a gold standard a test set of translations is. It has similarly been used in the NLP sentiment transfer literature (Li et al., 2018; Wu et al., 2019). The BLEU score is calculated as a weighted combination of the uni-, bi-, tri and- quad-gram overlap of the generated sentence with the gold standard. In this dissertation we use the SACREBLEU package (Post, 2018) in Python with default settings to calculate BLEU score. The specifics for the method used by Li et al. (2018) are not specified in their paper.

The BLEU score has been shown to have a correlation with human evaluations of the quality of a translation (Papineni et al., 2002). However, the BLEU score coupled with accuracy does not paint a full picture of the performance of sentiment transfer system, and thus it is also necessary to employ human evaluators to assess a subset of the generated sentences.

### 4.3.2 Human Evaluation

Mirroring the work of Li et al. (2018), the second criteria for model assessment comprises qualitative assessments of sentences on three aspects, namely sentence fluency, accuracy of sentiment transfer and preservation of the content of the sentence. These are referred to simply as fluency, sentiment and content going forward. These evaluations comprise having a set of reviewers rate a sample of the sentences transferred from one sentiment to another on these three dimensions using a Likert scale with scores from 1 to 5. The questions asked as well as the prompts can be found in Appendix A in this dissertation. For each of the models, the same 100 sentences are each sampled for the questionnaire. Note that the test sentences are generated in the following manner:

1. 100 sentences, 50 positive and 50 negative, are randomly sampled from the gold-standard parallel corpus test dataset of Li et al. (2018).

2. Given time and budget constraints, we were limited with the number of human evaluations that could be conducted. Because Li et al. (2018) show the human evaluations for their best performing model in the DRG paper, we sought to compare our best performing models with theirs on the human evaluations. For BART for the Yelp dataset we utilise a model that is trained with data that has been 40% and 50% noised and for the Amazon dataset, we utilise a model that is trained with data that has been noised 30% and 50%. The Yelp 40 and 50 models are chosen as the highest accuracy results are achieved for these two BART models along with BLEU scores that are closest to the DeleteOnly Yelp Saliency baseline. The Amazon 30 and 50 models are chosen to allow for direct comparisons with the Yelp results. For clarity, the Amazon 40 model is not included for human evaluations as the accuracy drops from the 40% noising to the 50% noising, thus making it less comparable with the Yelp noising which shows a positive linear relationship between noising levels and model sentiment transfer accuracy. These two models also generate sentences transferred from one sentiment to another with a BLEU score very similar to the DeleteOnly Amazon Saliency baseline. Due to the final stage of the BART training pipeline not requiring noised sentences, we are able to feed the original test sentences into the model whilst simply swapping the sentiment token at the beginning of the sentence to initiate sentiment transfer.

3. For the baseline DeleteOnly model (Li et al., 2018) it is necessary to first noise the sentences before feeding them to the model. We thus first remove words using the saliency method of the authors before feeding these sentences to the trained DeleteOnly version of the model at test time.

The evaluators were found by utilising the Amazon Mechanical Turk service. Each output in the test set was evaluated by three evaluators. In order for evaluators to be eligible for answering the survey about the generated sentences, it was necessary that they had completed 500 assignments prior to this, as well as achieving an acceptance score of 95% overall. However, to improve the calibre of the respondents, it was necessary to include a five-question quiz for the evaluators to complete before being allowed to access the full survey. Unfortunately, it was still necessary to

examine the results manually as it appeared that a number of the evaluators misunderstood the question regarding content preservation. In order to remove poor quality responses, the standard deviation of the ratings for the first content question for each sentence was calculated. Based on analysis, it was useful to further examine responses where the standard deviation of responses was greater than 1.5. This cutoff was chosen by examining the responses. For each of these responses, extreme scores of either 1 or 5 were reviewed in greater detail and when it was clear that the evaluator had not understood the question, the evaluator was excluded from further participation in the study and the questions re-entered into the question pool.

## 4.4 BART Training Curves

As noted in subsection 4.2.5, even though the first part of the BART training pipeline is trained with a DAE objective, the purpose of this part of the pipeline is to learn to transfer from one sentiment to another. Thus, the validation set is judged on the accuracy of the sentiment transfer from one sentiment to another and not simply reconstructing a sentence in the same sentiment. Training is stopped when the sentiment transfer accuracy on the validation set does not improve for three validation tests in a row. For each noised version of the datasets, these validation sets are conducted every 250 batches. The weights from the model from the highest scoring validation set are kept for the final model.

Figure 4.3 shows the training curves for the DAE and self-supervised training parts of the BART training pipeline for the Yelp dataset. These results represent only one of the three training runs for the BART models trained in this dissertation, however, the insights remain pertinent to the general training process. It is clear here that by increasing the degree of noising, the BART model improves the accuracy of the sentiment transfer. This is indicated by models trained on subsets of the data with a greater degree of noising having improved sentiment transfer accuracy. As these graphs show, the majority of the learning for sentiment transfer happens during the DAE training phase, with sharp rises in sentiment transfer accuracy occurring after the 2nd and 3rd validation steps at 500 and 750 batches respectively. The models achieve much smaller increases in sentiment transfer accuracy, however, not negligible, for the self-supervised phase of training. The highest results achieved for sentiment transfer for both the Yelp DAE and self-supervised models during training are shown in Table 4.4.

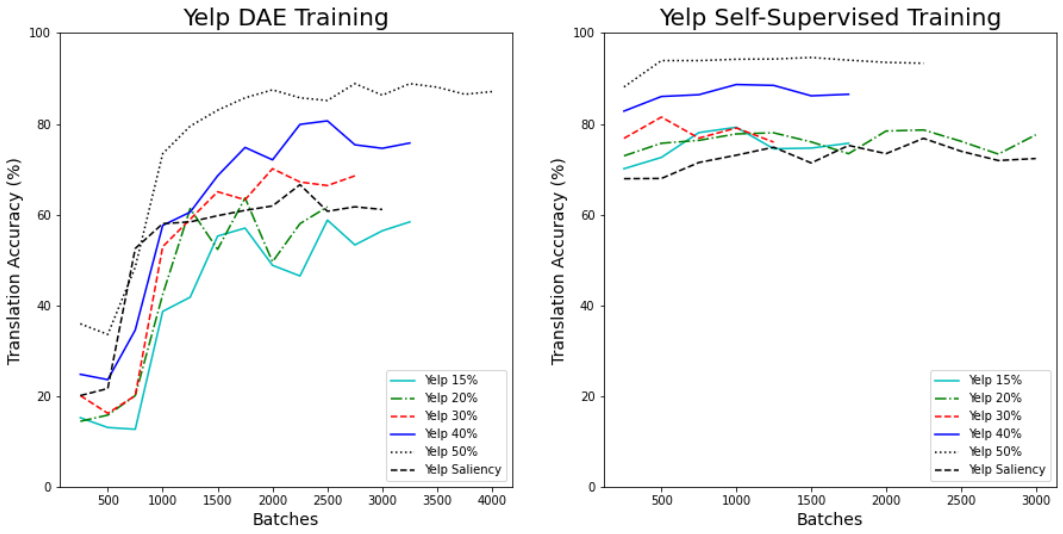


Figure 4.3: Training curves for the BART DAE and self-supervised training for Yelp. Self-supervised training leads to moderate gains in accuracy after DAE training. Greater noising leads to greater sentiment transfer accuracy.

It is worth highlighting that the sentiment transfer accuracy we achieve for the BART Yelp 50 DAE model is 88.9% (see Table 4.4). Even without the self-supervised training, this is higher than the sentiment transfer accuracy score we achieve for the BART Yelp 40 model *after* training on the self-supervised parallel dataset. This is also higher than the result we achieve for the DeleteOnly baseline for the saliency masked data. However, this result is based on the validation data and may be different for the test data. This is not tested in this dissertation as all comparisons on the test data are made from models trained across the entire training pipeline.

Figure 4.4 shows the training curves for the DAE and self-supervised training steps for the Amazon

| Input | DAE Final | Self-Sup Final |
|---|---|---|
| Yelp 15 | 58.8% | 79.2% |
| Yelp 20 | 63.7% | 78.6% |
| Yelp 30 | 70.1% | 81.5% |
| Yelp 40 | 80.6% | 88.7% |
| Yelp 50 | 88.9% | 94.6% |
| Yelp Saliency | 66.6% | 76.8% |

Table 4.4: Final BART model results chosen by validation on the sentiment transfer task for DAE and self-supervised pipeline training steps for each of the noised versions of the Yelp dataset

dataset. The training of all noising regimes, except for the saliency and 40% noised regimes, is stopped relatively quickly by the early stopping criterion during DAE training. However, on the right of Figure 4.4 we see that the subsets of the data, which train very quickly in the first step of the training pipeline, generally take longer to reach the early stopping criteria in the second stage of the training pipeline. This suggests that the model may have reached a local minimum in training in the first DAE step of the process that it moved out of in the second self-supervised stage of training. Similarly to the Yelp data, we see that as the masked proportion of the data increases, sentiment transfer accuracy improves. However, accuracy of the sentiment transfer is much lower with the Amazon data than it is with the Yelp data. Surprisingly, BART appears to learn to transfer sentiment from the saliency noised data primarily in the DAE part of training, with a drop seen in accuracy as the model moves to self-supervised training. In this run of the model, the self-supervised validation result is actually worse on the saliency noised data after self-supervised training when compared with the DAE training. The highest result achieved for sentiment transfer for both the DAE and self-supervised models during training for each of the levels of noising for Amazon is shown in Table 4.5.

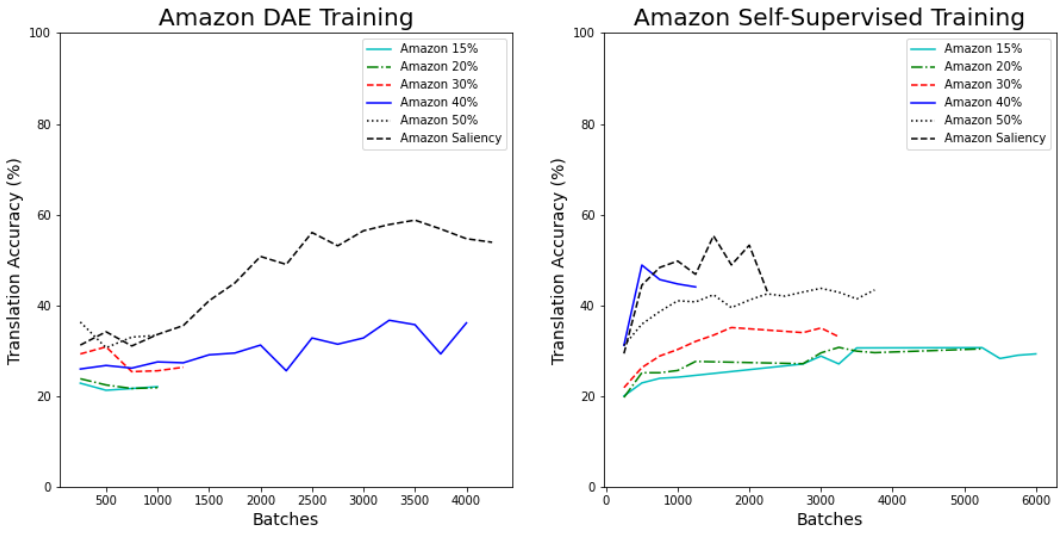


Figure 4.4: Training curves for the BART DAE and self-supervised training for Amazon. Self-supervised training leads to moderate gains in accuracy after DAE training, except for saliency noising. Greater noising leads to greater sentiment transfer accuracy.

| Input | DAE Final | Self-Sup Final |
|---|---|---|
| Amazon 15 | 22.85% | 30.68% |
| Amazon 20 | 23.83% | 30.75% |
| Amazon 30 | 30.86% | 35.12% |
| Amazon 40 | 36.72% | 48.88% |
| Amazon 50 | 36.33% | 43.75% |
| Amazon Saliency | 58.79% | 55.34% |

Table 4.5: Final BART model results chosen by validation on the sentiment transfer task for DAE and self-supervised pipeline training steps for each of the noised versions of the Amazon dataset

## 4.5 Chapter Summary

This chapter presented the details of the how each of the parts of the baseline and experimental models are set up in this dissertation. Along with describing how the models are trained, it also provided an overview of the hyperparameters we use to train the models. This was followed by a discussion of the automatic and human evaluations used, noting how the human evaluators were found on Amazon Mechanical Turk. This chapter then concluded with the training curves for the Yelp and Amazon datasets for BART. The next section provides a comparison of the results from the BART and baseline models and answers the research questions.

# CHAPTER 5

# RESULTS AND DISCUSSION

This chapter presents the results from the experimentation conducted in this dissertation. Whilst the previous chapter discussed training curves for the BART model, this chapter compares the results for the BART model and rationales noising against the baseline DeleteOnly model and saliency noising utilising both human and automatic evaluations.

Although a set of automatic evaluation metrics are used across both models and for varying degrees of noising the data, human evaluations of the data are utilised only for a subset of the trained models to compare them against previous approaches in the literature. This is primarily due to the cost and time constraints of using human evaluators.

## 5.1 Model Comparison

### 5.1.1 Automatic Evaluations

Table 5.1 provides a detailed overview of the results achieved in terms of accuracy and BLEU scores across the two models with different noising schemes. We see that the BART model achieves both higher accuracy and higher BLEU scores than the DeleteOnly model for the same level of noising with the Yelp dataset. This is true for all types of noising except the saliency method, where BART achieves a lower accuracy (-8.0%), but much higher BLEU score (+11.7 points) than the DeleteOnly model. As noted in subsection 4.2.3, the saliency-based noising achieves a mean proportion of 32.4% of the tokens masked, making it very similar to the 30% noised version of the

| | BART | | DeleteOnly | |
|---|---|---|---|---|
| Dataset | Classifier | BLEU | Classifier | BLEU |
| Yelp 15 | 70.3% | **29.6** | 49.0% | 24.3 |
| Yelp 20 | 74.5% | 27.5 | 69.2% | 16.1 |
| Yelp 30 | 78.6% | 26.5 | 72.0% | 17.8 |
| Yelp 40 | 83.5% | 19.3 | 81.2% | 13.6 |
| Yelp 50 | **93.7%** | 10.4 | 92.0% | 8.74 |
| Yelp Saliency | 74.2% | 26.8 | 82.2% | 15.1 |

Table 5.1: Classification accuracy and BLEU score of BART and baseline models for different levels of token noising for the Yelp dataset. **Bold** indicates best result.

data. This similarity appears more evident with the BART models where the Yelp 30 and Yelp Saliency models achieve similar accuracy and BLEU scores. However, in the case of the DeleteOnly model, the Yelp Saliency model has superior performance with sentiment transfer accuracy when compared with the Yelp 30, with an increase in 10.2%.

Table 5.2 presents the results of the accuracy and BLEU scores for both the BART and DeleteOnly models for the Amazon dataset. For Amazon, the saliency-based noising method shows a clear improvement in accuracy for the DeleteOnly and BART models when compared with the Amazon 30 results. It also shows a major improvement in BLEU score of 11.1 points for the BART model. The DeleteOnly Amazon 50 model achieves the highest accuracy of 58.8%, 11.4% above the DeleteOnly saliency masked baseline as shown in Table 5.2. However, this increase in accuracy comes at the cost of BLEU, with this model scoring a low BLEU score of 11.0, 11.3 points below our DeleteOnly saliency-based baseline. Whilst the Yelp results in Table 5.1 show that the BART model achieves higher accuracy and BLEU for the same level of noising when compared to DeleteOnly model for most noising schemes, this is not the case with the Amazon dataset. Here, for the same level of noising, the BART model achieves lower accuracy but higher BLEU for all versions of noising expect the saliency noising. Here, it achieves both higher accuracy and higher BLEU.

The low scores of the classifier for the automatic evaluations on the Amazon dataset may initially appear surprising, however, these accuracy and related BLEU scores are in line with the references that the original DeleteOnly model is compared against in Li et al. (2018).

Figure 5.1 provides a visual analysis of the results in Table 5.1. This diagram shows that there is a clear trade-off between content preservation - as measured by BLEU - and sentiment transfer

| | BART | | DeleteOnly | |
|---|---|---|---|---|
| Dataset | Classifier | BLEU | Classifier | BLEU |
| Amazon 15 | 28.5% | 36.4 | 39.2% | 19.5 |
| Amazon 20 | 31.4% | 32.4 | 40.0% | 29.6 |
| Amazon 30 | 35.2% | 25.8 | 42.4% | 22.3 |
| Amazon 40 | 48.1% | 22.1 | 51.7% | 16.4 |
| Amazon 50 | 44.7% | 17.3 | **58.8**% | 11.0 |
| Amazon Saliency | 53.6% | **36.9** | 47.4% | 21.3 |

Table 5.2: Classification accuracy and BLEU score of BART and baseline models for different levels of token noising for the Amazon dataset. **Bold** indicates best result.

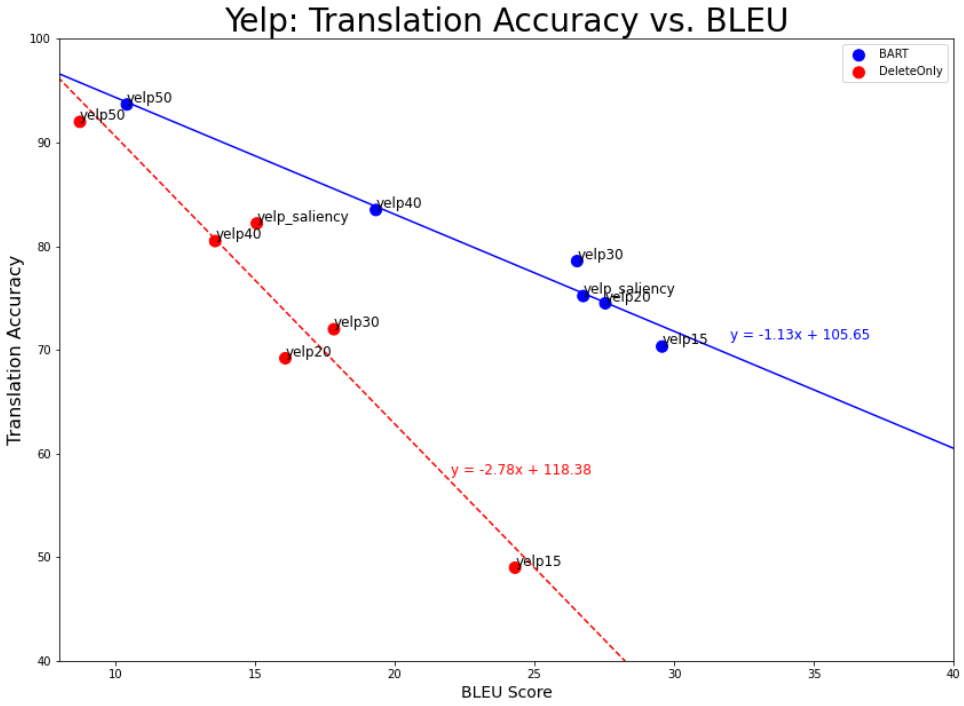


Figure 5.1: Graphical representation of accuracy vs. BLEU score for test outputs for BART and DeleteOnly baseline on Yelp dataset. Solid blue line represents BART and striped red line represents DeleteOnly. Smaller slope of the solid blue line indicates a more favourable trade-off between BLEU and accuracy for BART.

accuracy: the more accurate the sentiment transfer becomes, the less of the content of the original is captured by sentences transferred to the new sentiment. What is of interest in Figure 5.1 is that the ratio of accuracy to BLEU score is more favourable for the sentences generated from the BART model at test time. This is represented by the two trend lines for the BART and DeleteOnly results, with blue and red lines respectively. Here the slope of the line indicates the relative trade-off between BLEU score and accuracy, with a larger absolute number indicating a larger trade-off. For clarity, a larger trade-off indicates that gains to accuracy result in greater reductions in content preservation. The smaller absolute slope of the BART models' results point to a more favourable overall trade-off between accuracy and content preservation.

Figure 5.2 shows a graphical representation of the accuracy versus BLEU score for the BART and DeleteOnly models for different levels of noising for the Amazon dataset. Whilst the Yelp results

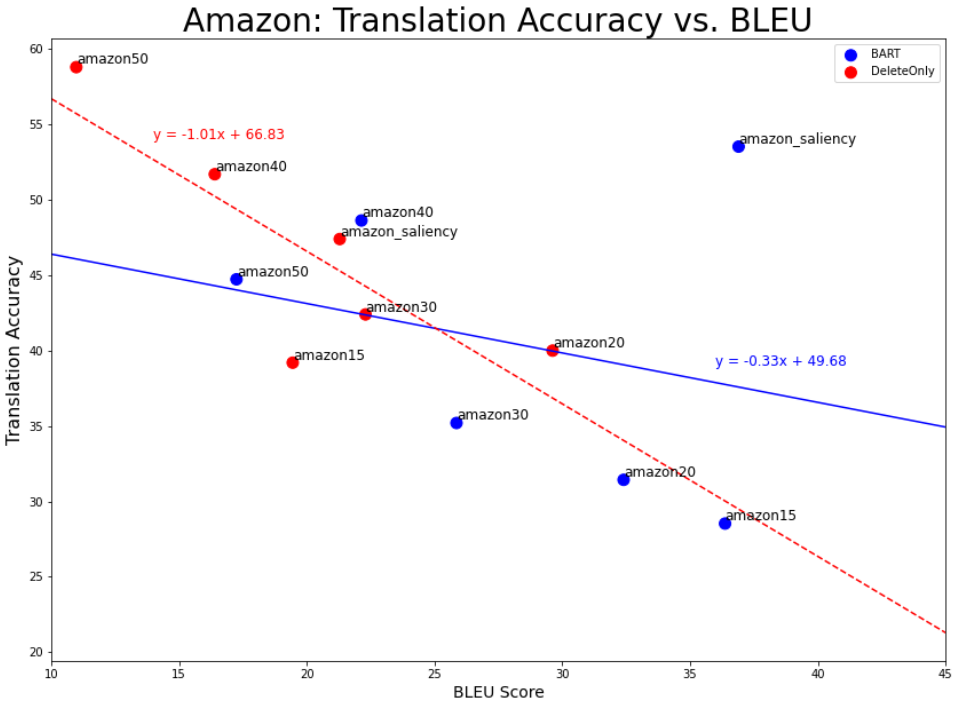


Figure 5.2: Graphical representation of accuracy vs. BLEU score for test outputs for BART and DeleteOnly baseline on Amazon dataset. Solid blue line represents BART and striped red line represents DeleteOnly. BART Amazon Saliency achieves the most favourable trade-off between accuracy and BLEU.

in Figure 5.1 show an advantage for the BART model with the trade-off between BLEU score and accuracy relative to the baseline model, these results are less clear in Figure 5.2. Although the slope of the trend line indicating the trade-off between BLEU score and accuracy is flatter for the BART model, this is primarily due to the influence of the BART Amazon Saliency model. When this saliency output is removed for both models, the trade-off between BLEU and accuracy was almost identical between the models (not shown here). However, the results from the saliency noising must be taken into consideration. This highlights that whilst it cannot be said that the BART model is necessarily better or worse than the DeleteOnly model for the Amazon data, when combined with the saliency noising, it achieves a significantly more favourable trade-off between content preservation and accuracy than the other models. BART Amazon Saliency also achieves the second highest accuracy score and the highest BLEU score for all of the Amazon models. It

appears therefore that the saliency noising on the Amazon dataset confers an advantage with the BART model. This may be due to this method of noising removing n-grams of up to four words and for the removal of contiguous phrases conferring an advantage to the BART model with the more difficult to classify Amazon data.

### 5.1.2 Human Evaluations

| Model | Noising | Content | Sentiment | Fluency |
|---|---|---|---|---|
| BART | Yelp 40 | **3.9** | 4.4 | **4.1** |
| BART | Yelp 50 | 3.2 | **4.5** | 4.1 |
| DRG | Yelp Saliency | 3.4 | 3.9 | 3.4 |

Table 5.3: Human evaluations for a subset of the Yelp generated sentences from BART and DeleteOnly. **Bold** indicates best result.

The results from the AMT human evaluations are shown in Table 5.3 for the Yelp results. Here the relationship between content preservation and sentiment accuracy appears less obvious than it does with the automatic evaluations. This is because a small jump in increased sentiment accuracy appears to be accompanied by a large drop in content preservation between the BART Yelp 40 and Yelp 50 models. This may be due to the accuracy difference of 5.6% between the Yelp 50 and Yelp 40 models being less perceptible to humans during the evaluation - given the average accuracy of 89.8% between the two models. However, the 8.5 point difference in BLEU scores may potentially be more perceptible to humans. The human evaluation for fluency, however, appears to be unrelated to the BLEU score. Given that the fluency score is the same for both BART models and different for the DRG model evaluated by the human evaluators, this suggests that the model type plays the biggest role in fluency score. By choosing the BART Yelp 40 model as the primary model of comparison, we can state that the BART model out-performs the DeleteOnly model across all human evaluations on the Yelp dataset.

| Model | Noising | Content | Sentiment | Fluency |
|---|---|---|---|---|
| BART | Amazon 30 | **3.4** | 3.4 | **3.9** |
| BART | Amazon 50 | 2.8 | **3.8** | **3.9** |
| DRG | Amazon Saliency | 3.3 | 3.2 | 3.5 |

Table 5.4: Human evaluations for a subset of the Amazon generated sentences from BART and DeleteOnly. **Bold** indicates best result.

The results from the AMT workers on the Amazon data are presented in Table 5.4. According to the human evaluations, the quality of the sentences generated from the BART models trained on the Amazon dataset are lower than they are from the BART models trained on the Yelp dataset. This is unsurprising given that the Amazon models score lower results on the automatic evaluation metrics. The gap between the human evaluations for BART and DeleteOnly is also smaller for the Amazon dataset than for Yelp. There also appears to be less of a strong relationship between the human evaluation and the automatic evaluation for the Amazon data. This is because the human evaluators give preferential evaluations for the sentiment of the BART Amazon 30 generated sentences vis-à-vis the DeleteOnly saliency-based sentences, in spite of the latter scoring higher results for sentiment transfer accuracy in the automatic evaluations. Although the greater degree of ambiguity in the sentiment of the Amazon sentences likely influences overall lower sentiment/accuracy evaluations, this suggests that higher fluency scores, may in turn, have an influence on higher sentiment scores. In the human evaluations on the Amazon data there is a clear trade-off between the scores for the content and the sentiment with the BART results. When comparing with the Yelp results, it is possible that the human evaluators are more sensitive to the accuracy of the sentiment when the accuracy is relatively low, as it is for the Amazon dataset. The difference in the accuracy and BLEU scores between the Amazon 30 and 50 noised versions of the data is similar to that of the Yelp 40 and 50 noised versions of the data. However, due to the difference in the average accuracy of these two subsets - 42.2% and 89.8% for Amazon and Yelp respectively - the same change in accuracy between the outputs may be viewed differently by the evaluators when the average sentiment transfer accuracy is low. One can draw similar conclusions about the human fluency ratings for the Amazon models as for the Yelp models. It appears that the model used - and not the BLEU or accuracy score - is the biggest factor influencing fluency score.

It is unsurprising that the BART model would outperform the fluency of the LSTM of the DeleteOnly model which is trained from scratch, given BART's extensive pretraining. Given the much larger corpus to which it is exposed, coupled with the larger model size, the pretrained model clearly better learns the conditional distributions of words in the training corpus. This is realised through more fluently generated sentences at test time. This may also be why the BART model achieves higher BLEU scores than the DeleteOnly model for the same level of noising.

Given that the models assessed using human evaluations are those which scored the highest accuracy

scores by automatic evaluation, one may argue that the human evaluations are biased towards generated data that maximised sentiment transfer accuracy, ignoring the impact of BLEU score. Although this may be the case, these models were chosen as they represented the best models from this dissertation when compared with the DRG models. Given that Li et al. (2018) present the results and hyperparameters of their best model in their paper, for the sake of comparability we compare our best model with theirs.

### 5.1.3 Comparisons with Transformers Baselines

This dissertation compares two different types of models against each other, two different noising schemes in the context of a single model and then discusses the two different noising schemes across models. For completeness, this section provides a very brief comparison with research conducted in sentiment transfer with two of the other Transformers (Vaswani et al., 2017) models discussed in subsection 2.3.2. As these models were not run during the course of this research, the results reported here are the results reported in the original papers.

Table 5.5 shows the results from comparing two different Transformers-based models with our BART models for each dataset. Although there are other Transformers models used in sentiment transfer, these two are the only ones with both human and automatic evaluations that are comparable to ours. For clarity, Sudhakar et al. (2019) utilised self-BLEU and a different method for human evaluations and Dai et al. (2019) utilise a different method for human evaluations. This prevents meaningful comparisons with ours and others' results.

| | Automatic Evaluation | | Human Evaluation | | |
|---|---|---|---|---|---|
| Model | Classifier | BLEU | Content | Sentiment | Fluency |
| **Yelp** | | | | | |
| BART Yelp 50 | 93.7% | 10.4 | 3.2 | **4.5** | 4.1 |
| BART Yelp 40 | 83.5% | 19.3 | 3.9 | 4.4 | 4.1 |
| Mask and Infill (Wu et al. 2019) | **97.3%** | 14.4 | **4.0** | 4.4 | **4.2** |
| FGIM (Wang et al. 2019) | 95.4% | **22.6** | 3.5 | 3.6 | 3.8 |
| **Amazon** | | | | | |
| BART Amazon 50 | 44.7% | 17.3 | 2.8 | 3.8 | 3.9 |
| BART Amazon 30 | 35.2% | 25.8 | 3.4 | 3.4 | 3.9 |
| Mask and Infill (Wu et al. 2019) | 84.5% | 28.5 | 4.1 | **4.0** | 4.0 |
| FGIM (Wang et al. 2019) | **85.3%** | **34.1** | **4.2** | **4.0** | **4.1** |

Table 5.5: Results of our model versus different transformers models. **Bold** indicates best result.

Although these results are not necessarily directly comparable given that we only quote the results provided in the original papers, they are still worthwhile for discussion. Table 5.5 highlights how BART Yelp 50 received the most favourable human evaluations for sentiment consistency. BART Yelp 40 also comes in a close second across all human evaluations, highlighting the strength of this model. The automatic evaluation of accuracy is not too far off the comparison papers, scoring only 3.6% below the highest accuracy. The BLEU results for Yelp 50 are, however, 12.2 points below the best scoring model (Wang et al., 2019). The results for the Amazon dataset are less favourable. Although human evaluations for fluency do not lag too far behind the comparisons, all other metrics are relatively far behind the best metrics achieved on the Amazon data.

Although all of these methods use the Transformer (Vaswani et al., 2017) architecture, their implementation is quite different. Wang et al. (2019) use an RNN on top of a Transformer-encoder and edit the latent space using a method called Fast Gradient Iterative Modification (FGIM). Wu et al. (2019) use a BERT model and learn to fill in masked words to reconstruct sentences in the required style. Unlike our method, both of these methods use a discriminator during training to guide their models.

## 5.2 Summary of Results

The results in this dissertation are in line with the results achieved in past papers (Li et al., 2018; Wu et al., 2019), where a greater degree of word deletion or masking - referred to as noising - leads to higher accuracy in sentence sentiment transfer. However, the trade-off as seen here and in previous work, is that the higher accuracy scores tend to be accompanied by lower content preservation scores. This is shown in greater detail in Figure 5.1 and Figure 5.2.

In terms of automatic evaluations, the BART model outperforms the DeleteOnly model on the Yelp dataset. Here the former model achieves greater accuracy and content preservation - as measured by BLEU score - for the same level of noising. However, the distinction is not so clear on the Amazon dataset, where it cannot be said that BART outperforms the DeleteOnly model. Table 5.2 shows that when compared with DeleteOnly on the Amazon dataset, BART achieves lower accuracy, but higher BLEU for the same level of rationales noising. This makes it challenging to state outright that one model is better than the other. The other challenge with interpreting

| | | Automatic Evaluation | | Human Evaluation | | |
|---|---|---|---|---|---|---|
| Model | Input | Classifier | BLEU | Content | Sentiment | Fluency |
| BART | Yelp 40 | **83.5%** | **19.3** | **3.9** | **4.4** | **4.1** |
| DeleteOnly | Yelp Saliency | 82.2% | 15.1 | 3.4 | 3.9 | 3.4 |
| BART | Amazon 30 | 35.2% | **25.8** | **3.4** | **3.4** | **3.9** |
| DeleteOnly | Amazon Saliency | **47.4%** | 21.3 | 3.3 | 3.2 | 3.5 |

Table 5.6: Best performing BART results with baseline results for Yelp and Amazon. **Bold** indicates best results for the specific dataset and metric.

the results is that in Figure 5.2 the BART Amazon Saliency model achieves a trade-off between accuracy and BLEU that appears out of sync with the other results. On the human evaluations, the BART model outperforms the DeleteOnly model on fluency, sentiment and content across both datasets.

Table 5.6 presents a comparison of the best BART models across both datasets when compared to the baseline DeleteOnly model. In the case of the Yelp dataset, the best model is the one which scored higher scores across all automatic and human evaluations vis-à-vis the baseline model. It is, however, less clear which BART model performed the "best" for the Amazon data. This is because we could not find a BART Amazon model which scores the best result across all metrics when compared with the baseline model. Thus, only one of the better performing Amazon BART models is shown here, however, the results for all of the models run were shown in section 5.1.

## 5.3 Qualitative Analysis

A selection of outputs of the test data are shown in Table 5.7. These highlight examples of the test data shown to the AMT participants to rate for the human evaluations in the dissertation. This also shows examples of the gold standard human sentiment transfer that our final outputs are compared against for the automatic evaluations.

In the first and last examples in Table 5.7, we see how the DeleteOnly model manages to transfer the sentiment of the sentence, however, the fluency of the sentiment transfer is worse when compared with the source sentence. Similarly, in these examples, whilst the BART Yelp 40 model manages to maintain the content of the sentences, the Yelp 50 and DeleteOnly models are less accurate on this dimension. Whilst this discrepancy between the results of the models is not always this acute,

| Data / Model | Output |
|---|---|
| Original | but was very disappointed with what actually arrived. |
| Reference Transfer | was very happy with what arrived. |
| BART Yelp 40 | everything was very good with what actually arrived. |
| BART Yelp 50 | but was very happy with what they did. |
| DelOnly Saliency | but it was very nice and easy to say that was dessert. |
| Original | i do not like the size of the dance floor. |
| Reference Transfer | i love the size of this dance floor! |
| BART Yelp 40 | i also really like the size of the dance floor. |
| BART Yelp 50 | i also really liked the feel of the dance floor. |
| DelOnly Saliency | i like the size of the dance floor. |
| Original | its quiet and nice people are here. |
| Reference Transfer | nice people are here, but it is too quiet and boring |
| BART Yelp 40 | its dirty and rude people are here. |
| BART Yelp 50 | its not like they are very busy. |
| DelOnly Saliency | its quiet people are here. |

Table 5.7: Original, reference sentiment transfer and final model at test time on the Yelp data

these examples highlight how the different results occurred for both the human and automatic evaluations of the Yelp outputs. These examples also highlight what it means for the content of the sentence to change during the sentiment transfer process. For example, in the first sentence we see that the Yelp 40 model maintains the core content - in this case the object of the sentence - "what actually arrived" remains intact. However, in Yelp 50 this changes to "what they did" and in the DeleteOnly model this is completely changed to "dessert". This was the general pattern that occurred as sentences received a greater level of noising.

Table 5.8 shows three examples of the final output from the three models that were used for the human evaluations for the Amazon data. Although the sentiment transfer of the first sentence seems to be relatively obvious, the correct sentiment transfer of the second and third sentences appears less obvious. This is predominantly due to the ambiguity of the sentiment of the original sentence in these latter examples. These sentences also show examples of how the content of a sentence can become corrupted during sentiment transfer, as is the case with all these DeleteOnly examples. This is also the case with the second and third sentiment transfers for the BART Amazon 50 examples.

| Data / Model | Output |
|---|---|
| Original | the cookbook that comes with it is adequate. |
| Reference Transfer | the cookbook that comes with it is terrible. |
| BART Amazon 30 | the cookbook that comes with it is terrible. |
| BART Amazon 50 | the cookbook that came with it was terrible. |
| DelOnly Saliency | the only good thing i ve used it is with it is adequate. |
| Original | it s almost like putting the phone into a high end pair of socks. |
| Reference Transfer | it s almost like putting the phone into a low end pair of socks. |
| BART Amazon 30 | it s not like putting the phone into a high end pair of shoes. |
| BART Amazon 50 | it s not even worth the effort into a single pair of socks. |
| DelOnly Saliency | it s almost like they are not into a high end pair of socks. |
| Original | so not that great for leaving on at night. |
| Reference Transfer | perfect for night |
| BART Amazon 30 | so far that works for leaving on at night.great product. |
| BART Amazon 50 | so far that works for me on at work. |
| DelOnly Saliency | so not that great for leaving a timer on. |

Table 5.8: Original, reference sentiment transfer and final model sentiment transfers at test time on the Amazon data

## 5.4 Answering The Research Questions

### 5.4.1 Research Question 1

First, we answer research question 1, namely whether using the method of rationales (Lei et al., 2016; Bastings et al., 2019) improves the ability of a model to transfer from one sentiment to another. This can be answered by comparing the best results from the method of rationales with the saliency-based noising with either of the models. One must take cognisance of the trade-off between accuracy and content preservation, where more noising generally leads to higher sentiment transfer accuracy, but lower content preservation. In light of this, one should compare noising methods that mask or delete a similar proportion of tokens in each sentence. In the case of the Yelp dataset, it is challenging to answer this question outright. When examining Table 4.2 we can see that the Yelp 30 noised version of the Yelp data is comparable with the saliency noised version of the data in terms of the mean proportion of masked or removed tokens per sentence. Coupled with Figure 5.1, we can see that these two noising methods achieve almost identical BLEU scores for the BART models. However, the accuracy results on the test set for the BART model are better for Yelp 30 (+4.4%) versus Yelp Saliency. However, for the DeleteOnly model this is reversed, where the saliency method achieves higher accuracy than Yelp 30 (+10.2%) but lower

BLEU (-2.7 points).

The Amazon 30 masked version of the data and the saliency noised version of the data have a similar mean level of noising (see Table 4.3). For a very similar level of noising, the saliency noised version of the data scores a higher accuracy score on sentiment transfer (+5.0%) and slightly worse on BLEU (-1.0 point) for the DeleteOnly Model when compared with Amazon 30. However, in the case of the BART model, the saliency masked dataset performs significantly higher on sentiment transfer (+18.4%) with a large jump in BLEU score (+11.1 points). Thus, for the BART model, the saliency noising method appears to confer a significant advantage in terms of both BLEU and accuracy score with the Amazon dataset. As noted earlier, this may be due to the saliency method removing contiguous words, given that it calculates the saliency for n-grams of up to four words. Given the more difficult to identify sentiment of the Amazon sentences, this may confer an advantage on the BART model given its extensive pretraining. For DeleteOnly the saliency method provides only a moderate increase in accuracy for a small drop in content preservation.

The saliency cutoff scores, coupled with the length of the n-grams used, were based on those from Li et al. (2018). Whilst this method may be computationally less intensive than the neural network-based rationales method (Bastings et al., 2019), it does require a greater degree of hyperparameter tuning when compared with the rationales method, which essentially has a single hyperparameter to tune: the expected proportion of sentiment tokens in corpus.

Another potential shortfall of the saliency method versus the rationales method is that, being rule-based, n-grams are either classified as positive or negative, without any consideration of context for the saliency method. Thus, one would expect the rationales method to consider more contextual information on a sentence-by-sentence basis when identifying sentiment words, leading to an increase in sentiment transfer accuracy for the same level of content preservation. However, this did not appear to be the case.

In answer to research question 1, it cannot be said that the method of rationales (Bastings et al., 2019) provides a significant improvement over the saliency method of Li et al. (2018) on the Yelp dataset. Although the results are better with BART on Yelp, this is not the case with DeleteOnly on Yelp. However, given that neither method is better than the other, we can state that the method of rationales is on par with the saliency approach and thus proves valuable on the Yelp dataset. In

the case of the Amazon dataset however, the saliency method appears to achieve a more favourable result for the same proportion of noising. This is especially true in the case of the BART model. This suggests that in cases where the sentiment of a sentence is less easy to identify, the saliency method may provide improved sentiment transfer accuracy and content preservation when used with a pretrained language model.

### 5.4.2 Research Question 2

Secondly, we answer research question 2, namely whether using a pretrained language model improves upon the ability of a model to transfer from one sentiment to another. In the case of the Yelp dataset, this is clearly the case. For almost all levels of noising, the BART model achieves both a higher accuracy and higher BLEU score, except for the saliency noising, where only the BLEU score is higher and the accuracy is lower when comparing BART with DeleteOnly. Along with this higher score, we also find a lower trade-off between accuracy and BLEU score as shown in Figure 5.1, further highlighting BART's out-performance of DeleteOnly on Yelp.

In the case of the Amazon dataset, the results are less obvious. Based on Table 5.2, we see that the BART model achieves a higher BLEU score, but lower accuracy for the same level of noising, except for the saliency noising. This suggests that although the pretrained model generates sentences transferred from one sentiment to another that are closer to the original in terms of content, it does so at the cost of accuracy, keeping sentence noising level constant. Although Figure 5.2 shows a lower trade-off between BLEU and accuracy for BART, this is largely influenced by the BART Amazon Saliency model which achieves results out of sync with the other models. This makes it challenging to meaningfully interpret this metric to compare BART and DeleteOnly.

Although fluency cannot be considered alone as a metric for how good a model is, one advantage of the pretrained models is that the larger model size and training dataset enable the models to produce language that is more fluent. This is evident across both datasets where the human evaluations for the fluency of the BART-generated data is higher in all cases. In this specific way, BART is unsurprisingly better than the DeleteOnly model based on human evaluations. In the case of Yelp, the superior human evaluations for BART agree with the automatic evaluations in relation to the DeleteOnly baseline. The models which achieved higher automatic evaluations also achieved higher ratings by human evaluators. However, for BART Amazon 30 the automatic evaluations are

lower for accuracy, as noted above, however, are rated more highly by the human annotators when compared with DeleteOnly. It is suggested here that the increase in the fluency may influence the human evaluations of the sentiment (accuracy) of these sentences transferred from one sentiment to another. According to human evaluations across both datasets the pretrained BART model, when coupled with the rationales noising, achieves superior results on content preservation, sentiment transfer and fluency for the Yelp 40 and Amazon 30 models when compared with the DRG baseline.

Thus, to answer research question 2, a pretrained language model leads to better sentiment transfer results on the Yelp dataset in terms of the automatic evaluations of accuracy and content preservation. However, the results are less clear on the Amazon dataset and no definitive answer was found. Unsurprisingly though, a pretrained language model does create more fluent sentences, as judged by human evaluators, which may inadvertently influence judgements on other metrics and thus lead humans to perceive BART as superior for sentiment transfer. Based on human evaluations, it was shown that the pretrained Transformer models achieve preferential qualitative evaluations for sentiment transfer.

## 5.5 Chapter Summary

This chapter has provided an overview and discussion of the results from the BART and baseline models trained with different levels of noising. We began with an in depth analysis of each model's performance at different levels of noising for each of the datasets. This was then followed by a discussion and comparison of the best performing BART models with their baseline counterparts.

In terms of automatic evaluation, it could not be said that either the saliency or rationale methods are superior to one another. It was shown that the BART model performs favourably against the baseline on the Yelp dataset in terms of overall accuracy and the trade-off between content preservation and sentiment transfer accuracy. However, it was not possible to say that BART outperformed the baseline for the Amazon dataset in terms of automatic evaluations.

The BART model was shown to outperform the baseline model across human evaluations, even in contradiction of the lower automatic evaluation scores on the Amazon dataset.

We shared an excerpt of the final model outputs as shown to the human evaluators. These were used to show concrete examples of the outputs given to the human evaluators to compare the models and

noising schemes. It also allowed for qualitative analysis of the functioning of the models. This was then followed by a brief comparison of our final results with two other Transformers-based papers. Finally, the results presented in this chapter were used to answer the two research questions stated in the introduction of this dissertation.

# CHAPTER 6

# CONCLUSION

This chapter presents a summary of the findings from Chapter 5, along with highlighting potential shortfalls and suggestions for future research.

## 6.1 Summary of Findings

This dissertation examines whether a specific method from the NLP interpretability literature, namely the method of rationales (Lei et al., 2016; Bastings et al., 2019), can be used to improve sentiment transfer. It also examines whether there is a benefit in using pretrained language models over a language model trained from scratch, particularly for the task of sentiment transfer. To answer these questions, we built a data processing and training pipeline using the implementation of the method of rationales of Bastings et al. (2019) and the BART Transformers model (Lewis et al., 2020). These results are compared against a heuristic data noising process coupled with the DeleteOnly version of the Delete Retrieve Generate model (Li et al., 2018).

The results in this dissertation are consistent with the results in other papers in the sentiment transfer literature (Wu et al., 2019; Li et al., 2018) whereby there exists an almost linear relationship between accuracy and BLEU score of a sentence transferred from one sentiment to another. This can be interpreted as saying the better able a model is to transfer from one sentiment to another, the less faithful the transferred sentence will be to the original sentence in terms of content. This is likely due to the model taking a greater degree of signal from the sentiment token prepended to a sentence when less tokens are available for sentence reconstruction during training. With increased dependence on the sentiment token, this improves sentiment transfer accuracy at the cost of content preservation. We see, however, that the BART model is able to achieve a more favourable trade-off between sentiment transfer accuracy and content preservation on the Yelp dataset than the DeleteOnly model.

When evaluating the benefit of the saliency method of sentiment word identification (Li et al., 2018) versus the neural network-based method of rationales (Bastings et al., 2019), it is unclear which method is better. This is compounded by the saliency method having different trade-offs between BART and DeleteOnly, thus it cannot be said that either method achieved outright better results

when controlling for the models used. On the Amazon dataset, when paired with the BART Model, the saliency method does achieve more favourable results than the other methods in terms of the BLEU-accuracy trade-off.

For Yelp, the automatic evaluations show a clear improvement in accuracy and BLEU scores of sentences transferred from one sentiment to another from the pretrained BART model versus the DeleteOnly model tuned from scratch. The automatic evaluations are, however, less clear for the Amazon dataset. In spite of these mixed automatic evaluations for both datasets, there is still a clear benefit to using a pretrained model for generation. This is shown by the higher human evaluations for the results from the pretrained model. This suggests that fluency ratings may have an inadvertent impact on other human ratings and that a more fluent model may be regarded as more preferable by human evaluators. This is the case even if sentiment transfer performance, as measured by automatic evaluations, does not match that of a less fluent model.

## 6.2 Recommendations for Future Research

- One of the primary challenges with the datasets utilised in this research is the ambiguity of the sentiment of many of the sentences in the Amazon corpus. This is particularly evident when reading the text. It would therefore be worthwhile to extend this study by utilising additional datasets such as the gender and age-based style dataset of Subramanian et al. (2018), albeit not public at the time of writing this dissertation. One could also experiment with selecting a subset of the Amazon data such that the classifier is more 'certain' of the sentiment of these sentences. This would imply that the sentiment of these sentences is more defined and would thus be more easy to change to a discretely different sentiment.
- The self-supervised training pipeline was created due to a failed experiment with using an initial adversarial model in this dissertation. Initially, our BART model was trained with oscillating DAE and sentiment transfer objectives. First the model took a noised sentence and reconstructed it in the original sentiment, conditioned on the sentiment token. Following this, the sentiment token was swapped and the model aimed to transfer the noised sentence into the sentiment specified by the sentiment token. To generate discrete tokens that could be passed to a pretrained classifier for classifying the generated sentences, we used the Straight

Through Gumbel Softmax trick (Jang et al., 2016). This enabled us to generate discrete outputs (tokens) on the forward pass whilst still enabling continuous gradients for training on the backwards pass. The error of sentence sentiment in the reconstruction - akin to the DAE part of our pipeline - and the sentiment transfer phases were passed back through the BART model from the classifier. This method is similar to that employed by Wu et al. (2019). Unfortunately, in spite of experimenting with a large array of hyperparameters, this approach was unsuccessful. The outputs from this experimentation 'collapsed', with the model generating sentences that comprised repetitions of the same word, either from the positive or negative sentiment. Thus, this research could be improved upon by utilising an adversarial model where the language generating model is taught to generate sentences in a specific sentiment by interacting with a discriminator-classifier.

- At the commencement of research, BART (Lewis et al., 2020) had recently been released, however, subsequent to this, a number of new encoder-decoder transformers have been released. Thus, it may prove valuable to implement the research in this dissertation utilising a more recent encoder-decoder model, such as T5 (Raffel et al., 2020). T5 is suggested because, with 11 billion parameters and pretrained with 750GB of text, it is likely to improve upon the smaller BART (Lewis et al., 2020) model comprising only 139 million parameters.
- As machine learning models are employed in more areas to enhance decision making, interpretability becomes more important in their functioning. At a high-level, this dissertation has explored utilising a method from the interpretability literature in sentiment transfer. As this field of inquiry further matures, additional methods from this domain could be utilised in the NLP style transfer domain, for example by utilising more recent interpretability methods to identify and mask or delete style words.
- One benefit of the saliency-based noising is that it appears to have a higher degree of variance in the proportion of tokens that it selects as sentiment-specific tokens. This is considered an advantage because it may be indicative of the heuristic allowing for more adaptability in the number of words masked or removed per sentence. Another benefit may be that the saliency noising selects n-grams of up to four tokens for removal or masking. This suggests that a fertile area for future research may be to experiment with noising methods that allow for more

variability in the degree or proportion of word removal or masking. It may also be valuable to further explore the rationales method, however, allowing the model to mask contiguous sequences of tokens.

# REFERENCES


Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. *ICLR* (2015).

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable Neural Predictions with Differentiable Binary Variables. In *ACL (1)*. Association for Computational Linguistics, 2963–2977.

Anton Chernyavskiy, Dmitry Ilvovsky, and Preslav Nakov. 2021. Transformers: "The End of History" for NLP? *arXiv e-prints* (2021), arXiv–2105.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*. Association for Computational Linguistics, 1724–1734.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuan-Jing Huang. 2019. Style Transformer: Unpaired Text Style Transfer without Disentangled Latent Representation. In *ACL*. 5997–6007.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT (1)*. Association for Computational Linguistics, 4171–4186.

Cícero Nogueira dos Santos, Igor Melnyk, and Inkit Padhi. 2018. Fighting Offensive Language on Social Media with Unsupervised Text Style Transfer. *CoRR* abs/1805.07685 (2018).

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style Transfer in Text: Exploration and Evaluation. *Association for the Advancement of Artificial Intelligence (AAAI)* (2018), 663–670.

Philip Gage. 1994. A new algorithm for data compression. *C Users Journal* 12, 2 (1994), 23–38.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. A Probabilistic Formulation of Unsupervised Text Style Transfer. In *ICLR*. OpenReview.net.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th international conference on world wide web*. 507–517.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. In *ACL (1)*. Association for Computational Linguistics, 328–339.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An Introduction to Statistical Learning: with Applications in R*. Springer.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016).

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A Conditional Transformer Language Model for Controllable Generation. arXiv:1909.05858 [cs.CL]

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *ICLR*.

Ponnambalam Kumaraswamy. 1980. A generalized probability density function for double-bounded random processes. *Journal of hydrology* 46, 1-2 (1980), 79–88.

Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2016. Rationalizing Neural Predictions. In *EMNLP*. The Association for Computational Linguistics, 107–117.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *ACL*. Association for Computational Linguistics, 7871–7880.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, Retrieve, Generate: a Simple Approach to Sentiment and Style Transfer. In *NAACL-HLT*. Association for Computational Linguistics, 1865–1874.

Lajanugen Logeswaran, Honglak Lee, and Samy Bengio. 2018. Content preserving text generation with attribute controls. In *Advances in Neural Information Processing Systems*. 5108–5118.

Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 4765–4774.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *EMNLP*. The Association for Computational Linguistics, 1412–1421.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR (Workshop Poster)*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*. Association for Computational Linguistics, 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *NAACL-HLT*. Association for Computational Linguistics, 2227–2237.

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *WMT*. Association for Computational Linguistics, 186–191.

Ofir Press and Lior Wolf. 2017. Using the Output Embedding to Improve Language Models. In *EACL (2)*. Association for Computational Linguistics, 157–163.

Reid Pryzant, Diehl Martinez Richard, Nathan Dass, Sadao Kurohashi, Dan Jurafsky, and Diyi Yang. 2020. Automatically Neutralizing Subjective Bias in Text. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. (2018).

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. (2019).

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67.

Sudha Rao and Joel R. Tetreault. 2018. Dear Sir or Madam, May I Introduce the GYAFC Dataset: Corpus, Benchmarks and Metrics for Formality Style Transfer. In *NAACL-HLT*. Association for Computational Linguistics, 129–140.

Sebastian Ruder. 2017. Transfer Learning - Machine Learning's Next Frontier. `http://ruder.io/transfer-learning/`.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *ACL (1)*. The Association for Computer Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2017. Style Transfer from Non-Parallel Text by Cross-Alignment. In *Advances in Neural Information Processing Systems*. 6830–6841.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning Important Features Through Propagating Activation Differences. In *ICML (Proceedings of Machine Learning Research, Vol. 70)*. PMLR, 3145–3153.

Sandeep Subramanian, Guillaume Lample, Eric Michael Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2018. Multiple-attribute text style transfer. *arXiv preprint arXiv:1811.00552* (2018).

Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. 2019. "Transforming" Delete, Retrieve, Generate Approach for Controlled Text Style Transfer. In *EMNLP/IJCNLP (1)*. Association for Computational Linguistics, 3267–3277.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*. PMLR, 3319–3328.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*. 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing*. 5998–6008.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. 1096–1103.

Ke Wang, Hang Hua, and Xiaojun Wan. 2019. Controllable Unsupervised Text Attribute Transfer via Editing Entangled Latent Representation. In *Advances in Neural Information Processing Systems*. 11034–11044.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *EMNLP (Demos)*. Association for Computational Linguistics, 38–45.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149* (2019).

Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. "Mask and Infill": Applying Masked Language Model to Sentiment Transfer. *arXiv preprint arXiv:1908.08039* (2019).

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, and Yuan Cao. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. (2016). arXiv:1609.08144 [cs.CL]

Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image Captioning With Semantic Attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-Scale Generative Pre-training for Conversational Response Generation. In *ACL*. Association for Computational Linguistics, 270–278.

# APPENDIX A

# HUMAN EVALUATIONS QUESTIONNAIRE

Figures A.1 to A.4 present the four parts of the questionnaire that was administered to Amazon Mechanical Turk (AMT) workers on the AMT platform. For each Human Intelligence Task, called a HIT and represented by a questionnaire, each AMT worker was asked to provide three ratings for three outputs from the final model. The figures here give an example of the three ratings - each for a different qualitative aspect of the output - that each worker was asked to provide on a single sentence.

Each question was accompanied by a collapsible panel, shown open in these examples, to provide the evaluators with examples to clarify what each of the questions was querying. The order of the sentences were randomly shuffled for each HIT prior to creating the questionnaire. This means that for Yelp the first HIT may have been ordered with outputs like [BART Yelp 40, DeleteOnly Yelp Saliency, BART Yelp 50] and the second HIT ordered like [DeleteOnly Yelp Saliency, BART Yelp 50, BART Yelp 40] with similar shuffling for the remaining HITs. Each generated output sentence was evaluated by three AMT workers with 50 positive and negative reference sentences evaluated for each of the final models.

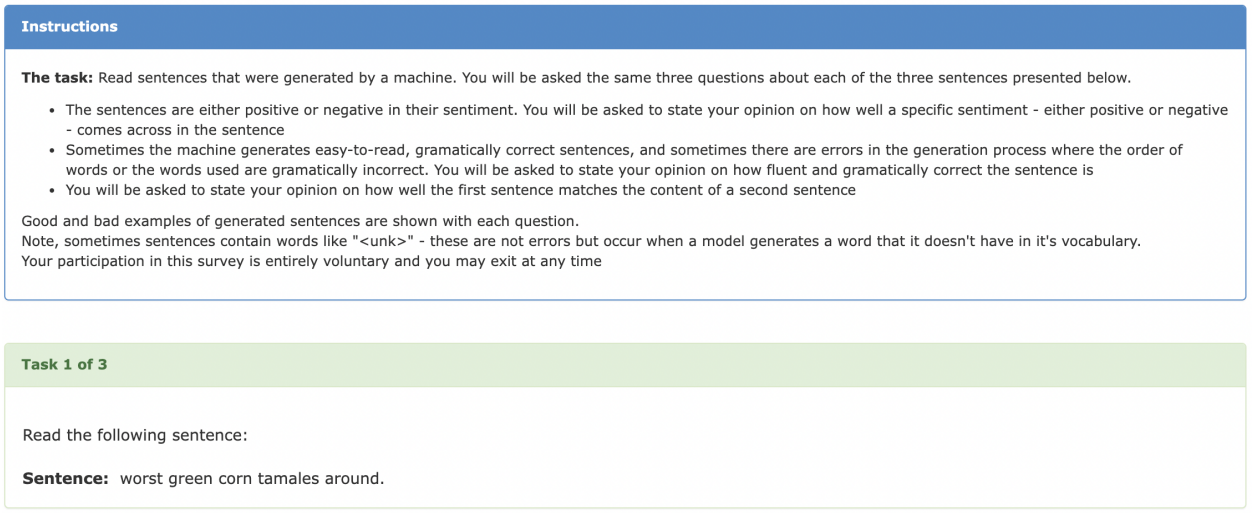


Figure A.1: The instructions provided for answering the questionnaire on AMT

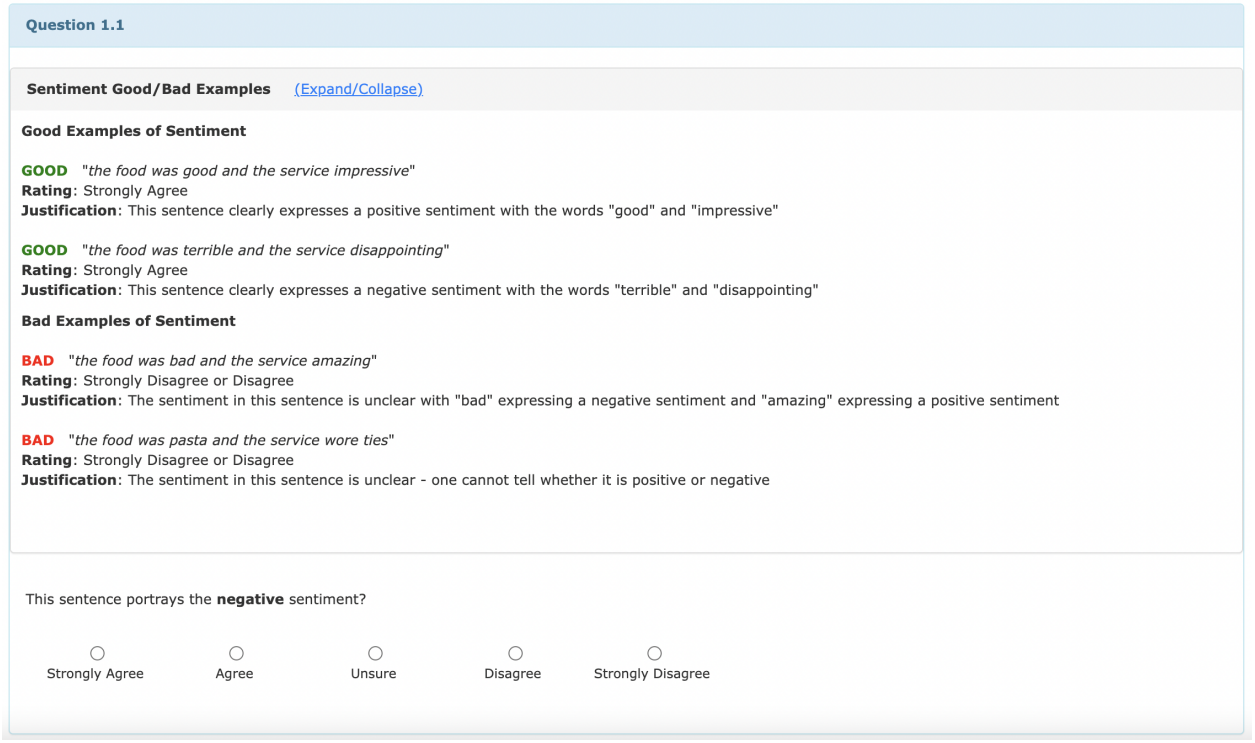


Figure A.2: Question and example for querying the sentiment of a sentence transferred from one sentiment to another with human evaluators

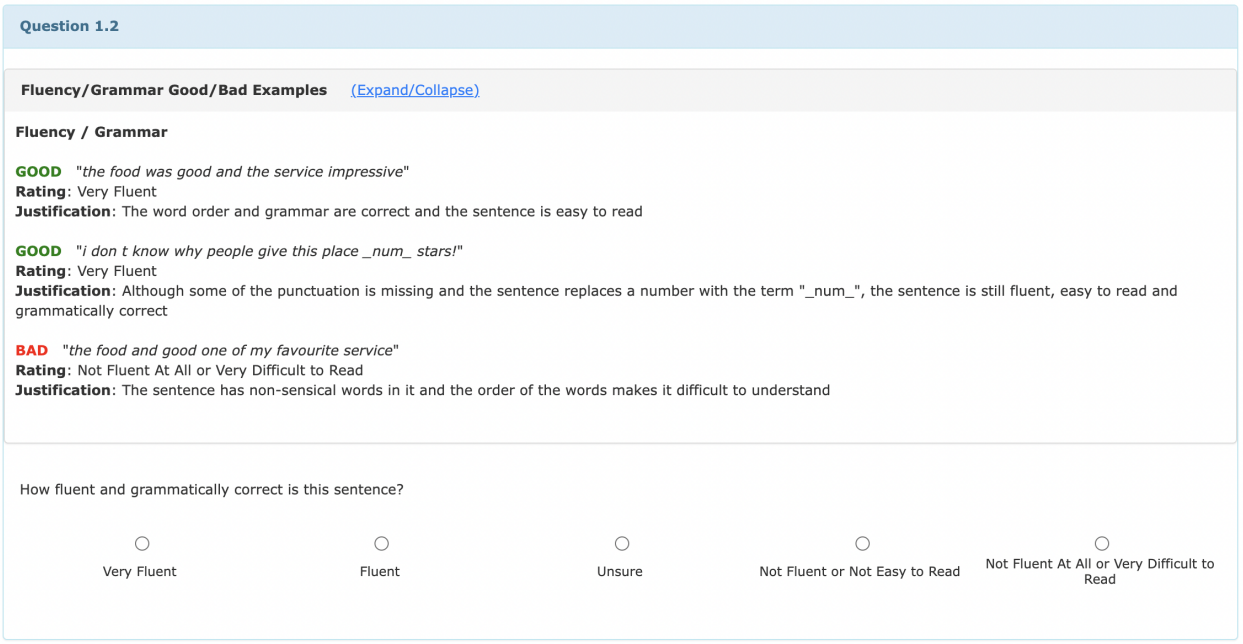


Figure A.3: Question and example for querying the fluency of a sentence transferred from one sentiment to another with human evaluators

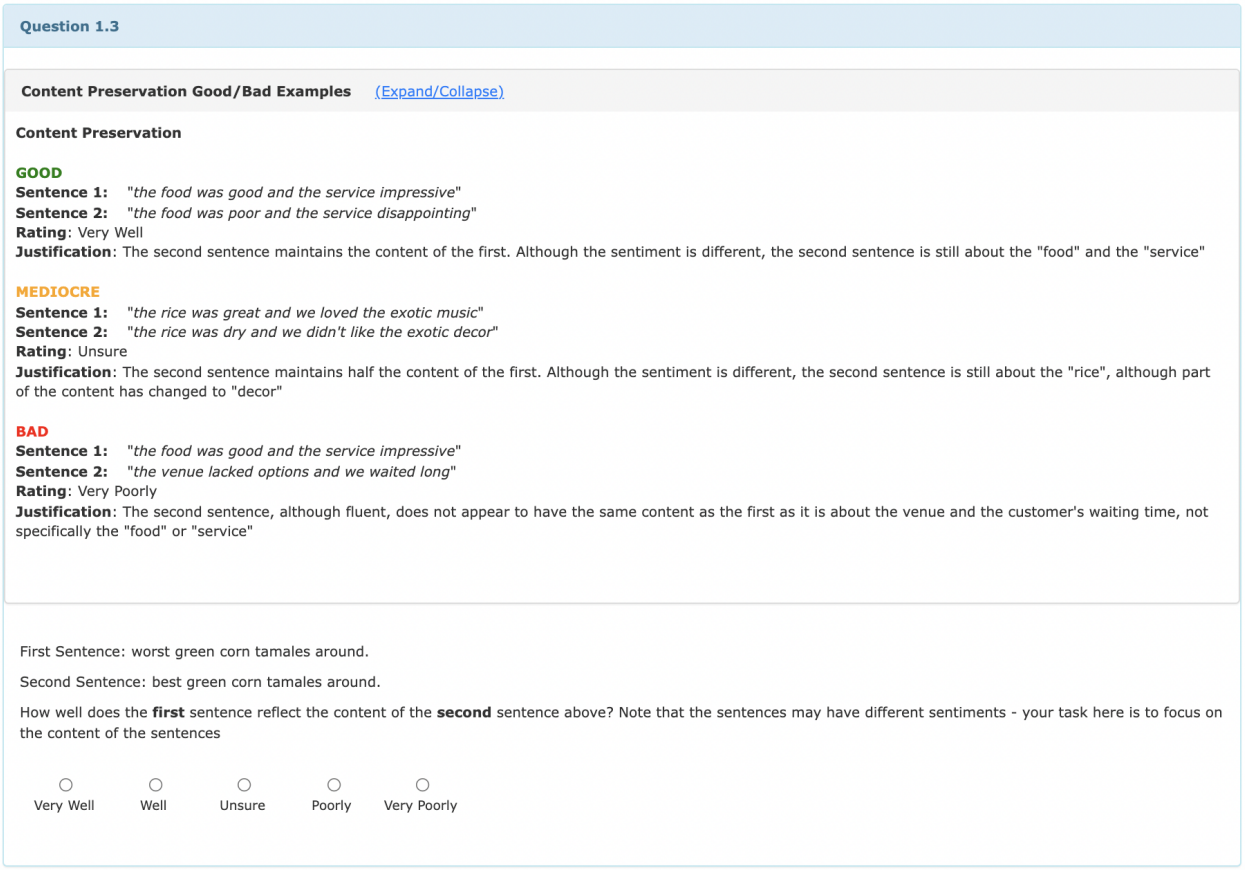


Figure A.4: Question and example for querying the content preservation of a sentence transferred from one sentiment to another versus the source sentence with human evaluators