A Generative-Discriminative Framework for Title Generation in the E-commerce Domain

Chuan-Yi Chang

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2018

Reading Committee:

Marco Guerini, Chair

Matteo Negri

José GC de Souza

Gina-Anne Levow

Program Authorized to Offer Degree:
Department of Linguistics

University of Washington

Abstract

A Generative-Discriminative Framework for Title Generation in the E-commerce Domain

Chuan-Yi Chang

Chair of the Supervisory Committee:
Dr. Marco Guerini
Department of Chair

Multi-Engine (system combination) Machine Translation (MEMT) systems produce several outputs which will then be refined to produce a higher quality translation output. In the case where source sentences are absent in the pipeline, the task of post-editing will be even more challenging. This is then analogous to finding a consensus token sequence of user-input source sentences, in order to remove any potential errors.

In a similar setting, major e-Commerce sites contain millions of user-created titles for a large number of products. These titles are similar in nature to translation outputs from different machine translation systems. Listed in these sites are products that have individually numerous source titles created by different sellers. Target product titles could serve as an effective way with which collections of products can be clustered and displayed. At the same time, the continuous upload of user-created titles calls for the need for new target product titles on a regular basis. Since the creation of effective target titles poses a challenge, many e-Commerce sites resort to manual curation of titles. However, the process of creating target titles would amount to great costs since manual construction of these titles is both a costly process and one that is never-ending.

On the other hand, selection of the target product title from the existing seller-created source titles would subject the title to potential errors in the title. Therefore, the idea of using a source titles as a replacement of the ideal title would be futile. In light of the above

scenarios, we aim to investigate a robust system that takes as input multiple user-input source strings with potential errors, and generate a single high-quality target string. More concretely, our research is performed on collected source title sets coming from the domain of e-Commerce, in order to mimic the generation of high-quality human-created target product title.

Thus, this thesis focuses on the idea of creating target titles given existing source titles. We propose a Generative-Discriminative framework which consists of two main components: (i) a generative feature-rich decoder that transforms the initial source title search space into one with better quality generated titles, and (ii) a discriminative attention-based re-scorer that selects the best title among the generated ones to be the predicted target title. For the practical need of the automatic generation, we propose a robust approach that could generalize the supervised models to large unseen datasets. Moreover, we tested our systems in a bi-lingual setting and found that it performed competitively in both German and English.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY

SOURCE TITLE: an error-prone user-created sequence of words that describes an object.

SOURCE TITLE SET: a set of source titles.

SOURCE TITLE SET SIZE: number of source titles in a set.

REFERENCE TITLE: a perfect human-created title for an object.

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to Fondazione Bruno Kassler (FBK) and eBay Inc., where he has had the opportunity to work with the Dr Marcello Federico, Dr. Marco Guerini, Dr. Marco Turchi, Matteo Negri, Dr. Evgeny Matusov and Dr. José GC de Souza. Without whose invaluable help and patience, I would not have been able to have this opportunity to work on this project, and be able to receive so many great advices.

Moreover, I want to thank Mohammad Amin Farajian, Surafel Melaku Lakew, Mattia Antonino Di Gangi, Anna Feltracco, Duygu Ataman, Enrica Troiano, Oscar Araque, Prashant Mathur, Rajen Chatterjee, and Simone Magnolini for coming into my life and offering advices to life and to areas related to my research. For without all the discussions I have had with them, this work could have been more primitive than its current state.

Last but not the least, I want to thank all the beautiful lakes and mountains in Trento, Italy. Their proximity and beauty served as the perpetual, natural encouragements to my time there.

DEDICATION

to my family

Chapter 1
INTRODUCTION

1.1 Motivation

In numerous NLP-related industrial setting there exist needs for refinement of system outputs. One prominent example of this is the Multi-Engine (system combination) Machine Translation (MEMT) where system outputs are recombined to form a consensus output so as to remove erroneous parts. While MEMT handles it jointly by taking into account source sentences, there are times when even the source sentences are missing in the pipeline process.

Other than MEMT, e-Commerce sites are dealing with similar issues with regard to product titles. As hundred of millions of shoppers daily frequent these sites to purchase goods of all kinds, many become reliant on these sites for their purchases. As such, User Interface (UI) of search result pages is becoming crucial for the user experiences. More specifically, e-Commerce take as input seller-created source titles, and there arises a need to create target title.

For example, a given product such as a projector can have different types and would be sold by multiple sellers. Each of these sellers can create a different source title, resulting in a noisy search page. One specific example is shown in Figure 1.1, where different types of projectors belonging to the brand *Panasonic* are available.

This serves to enhance user experience since source titles could then be "clustered" based on their types, thus showing on the search result page only titles where each title leads to a set containing all the relevant items. This is shown in 1.2. As such, we intend to propose a system that could summarize and generate a error-free title that could group source titles of the same type together by constructing the target title as shown in 1.2, where it says "*Panasonic PT-AE8000U LCD Projector*".

Figure 1.1: Example source title set and their representations as **black circles** in the space of titles

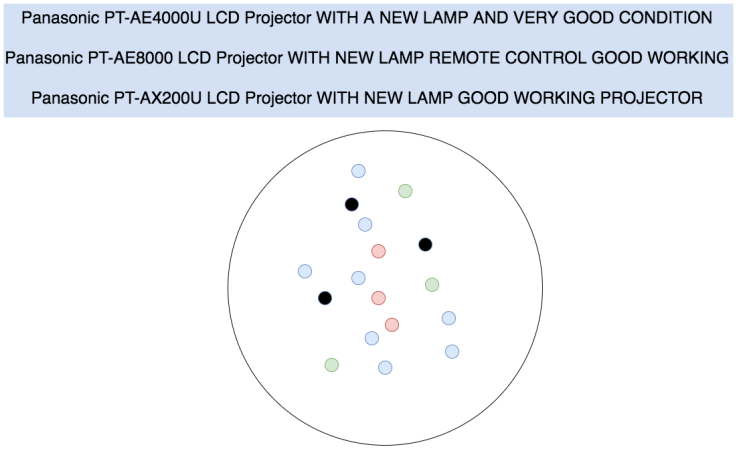
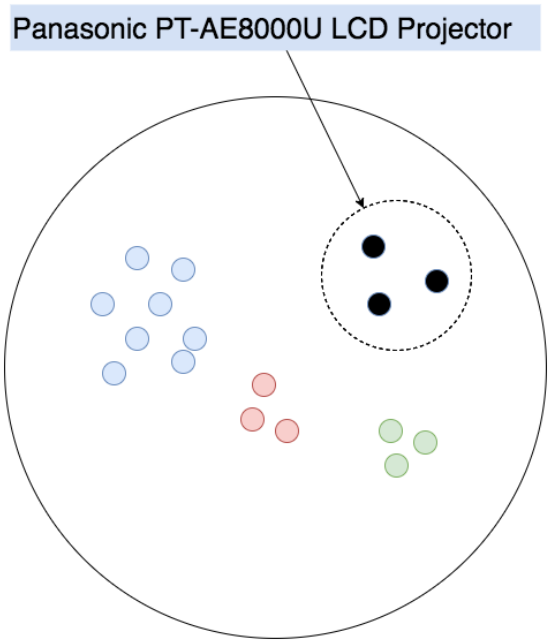


Figure 1.2: Generated title for clustering of titles.



We refer to the above problem as Title Generation (TG) since we attempt to research on this issue in the specific domain of e-Commerce. The goal is fundamentally different from MEMT in that source sentences are not provided, but merely the system outputs which comes in the form of user-generated titles in our task. On the other hand, the selection of a title from the existing source titles would subject the title creation process to two major drawbacks:

- Biases for users whose titles are used as the target title, since selected title gives that particular user the added Search Engine Optimization advantage in getting noticed.
- Human-created titles may provide redundant, misleading information. E.g. "*VERY GOOD CONDITION*","*best ever*" etc.

As such, the goal of this thesis is to propose a model that will automatically generate target titles given a set of source titles. For practical reasons, we also want to emphasize on the computational tractability of the proposed models. Generation process should happen within a reasonable time frame. We also intend to ensure that our model could be trained so that it is easily generalizable to unseen data without re-training.

1.2 Task Definition

1.2.1 High-level View

Title Generation (TG) is defined as the process where the function $TG(T)$ takes as input a set of source titles $T = \{t_1, t_2, ..., t_n\}$, and results in the target title t_p. A title t_p has to fulfill the following requirements:

1. Error removal: t_p must not contain user-specific tokens or phrases.
2. Readability: t_p should follow the natural ordering of a human-created source titles for fluency. For instance in the case of our data, one possible natural order could be: brand, item, serial number.

The main objective of the task is to provide a system that could mimic the reference titles, given an input of numerous titles.

1.2.2 Problem Formulation

Each source title set T contains a set of source titles $t_i \in T$, which consists of a sequence of tokens w_i. Mathematically, each title can be seen as the sequence $t_i = \{w_1^i, w_2^i, ..., w_m^i\}$. The probability of a sequence of tokens can always be decomposed using the chain rule (without assumptions):

$$\begin{aligned} p(t_i) &= p(w_1^i, \ldots, w_m^i) \\ &= p(w_1^i) \prod_{j=2}^{m} p(w_j^i | w_1^i, \ldots, w_{j-1}^i) \\ &= p(w_1^i) \prod_{j=2}^{m} p(w_j^i | h_j) \end{aligned} \tag{1.1}$$

Here, we will use $h_i = \{w_1^i, \ldots, w_{j-1}^i\}$ to denote the history of the j-th word w_j. As such, there exists a natural ordering of tokens as given by configuration d, as represented by the language modeling derived from source title set T:

$$p(w_1, \ldots, w_m) = p(w_1) \prod_{i=2}^{m} p(w_i | w_1^{i-1})$$

We also define desired title length as D and generate t_p within the constraints of all the given tokens $w_i \in T$, where $t_p = \{w_1^i, \ldots, w_D^i\}$. Thus without loss of generality, we want to generate the title t_p that maximizes the probability of both selected token set and the token ordering set, given the source titles T and outside knowledge O that is external to the source title set:

$$\begin{aligned} t_p &= \arg\max_{v \in \mathcal{P}(V)} \left(\arg\max_{d} P(v^d | T, O) \right) \\ &= \arg\max_{v \in \mathcal{P}(V), d} P(v^d | T, O) \end{aligned} \tag{1.2}$$

where:

$d(\cdot)$ is the token ordering configuration, it places the tokens in a specific order.

T is the source title set.

O is the external knowledge.

$\mathcal{P}(V)$ is the power set of vocabulary V.

As we can see, this formulation concerns two particular aspects of TG. They are

1. Token selection, $v \in \mathcal{P}(V)$.

2. Token ordering, $d \in D$, where D is all possible orderings of token $w^i \in v$.

To further understand the task, we will use an example from the dataset we used in our experiments. For each set of source titles, this dataset contains: (i) several source titles and (ii) a human-curated title that describes the product being referred to (details are given in Chapter 4).

Given that the human-curated title for the following source title set is

Reference: *Panasonic PT-AE3000U LCD Projector*

The source titles associated with this title are the following (the non-overlapping phrases are in bold):

1. *Panasonic PT-AE3000U LCD Projector* ***WITH NEW LAMP AND VERY GOOD CONDITION***

2. *PANASONIC PT-AE3000U Projector* ***Replacement Lamp with Housing***

3. *Panasonic PT-AE3000U PROJECTOR with remote and extra bulb*

4. *Quality Compatible Panasonic PT-AE3000U* ***165 Watt 2000-Hrs HS***

5. *Panasonic PT-AE3000U PROJECTOR,* ***remote,extra bulb, and INVIDEO HDMI switcher***

6. *PT-AE3000U LCD 1080P Proj 60K:1 1600 Lumens HDmi/component*

7. *PANASONIC Projector Replacement Lamp with Housing*

This example shows that the ideal title would involve selection of tokens in the process of combination. Since phrases like *VERY GOOD CONDITION* are not relevant to the product itself, a major component of the task involves reducing the likelihood of the inclusion of these redundant tokens. Moreover, we can see that most of these titles contain similar token orderings as that of the reference title, thus the need to transpose tokens is in fact not that ideal.

1.3 Task Scenario

For both theoretical and practical reasons, we intend to position the problem with some specific constraints which we hope to explicitly elucidate before the full-blown model exposition. We list them as below:

1. Computational Tractability: models should not be computationally intractable, and should complete title generation within a polynomial time as a function of the source title set's size.

2. Multi-lingual Settings: models should be able to perform competitively in multi-lingual scenarios, i.e., trained on some language A's training data, but capable to work similarly well on language B's testing data. In our case, we intend to test on bi-lingual data consisting of both English and German data.

3. Limited training data: only a limited subset of training data is given, and the models have to be able to generalize to larger unseen test data.

4. Robustness: the model should be ale to perform similarly well in different settings as given by the source title sets. E.g. varying sizes of source title set and varying average of title lengths.

We shall now discuss some of the existing perspectives that are relevant to the task.

1.4 System Combination (Multi-Engine Machine Translation)

Machine Translation (MT) is the task of converting text from a source language to a target language. Over the years, numerous successful MT approaches emerged [50, 94, 2].

There then exists a need to combine the different successful systems that performs competitively against each other into the consensus translation. As such, system combination could make the best out of these systems, focusing on the strengths while attempting to avoid the disadvantage of these systems. System combination works by either 1) selection from among the best translation hypotheses, or by 2) combining and generating new hypotheses based on the candidate hypotheses set [59].

In this light, TG is identical to system combination in the sense that both tasks receive an N-best set of candidate hypotheses to be re-combined into a new best hypothesis that is better in essence than all of the original candidates. However, in relation to this thesis, since the motivation for the work is to re-generate new titles based on the source titles, we will focus our efforts on the discussion of the latter − to generate new hypotheses based on the candidate source title set.

Following the classification by [59], we will first examine the system combination approaches through the lens of units of combination − token, phrase, and sentence.

1.4.1 Token-based Approaches

As was pointed out in[59], most of these token-based approaches include the use of the confusion network decoding model in which multiple sequences are first aligned, and then token sequences are re-ordered before finally being used for confusion network creation [63].

All candidate hypotheses are transformed into separate confusion networks before finally being summed up into a union of confusion networks by adding the probabilities of the arcs. However, while the process is considering all hypotheses as the primary hypothesis E_m, there are two major caveats to this approach. First there is a lack of consideration for long term dependencies between tokens in the paths. This makes it prone to break the sentence coherence and also render it a difficult task to account for the title semantics in units larger than a single word [59]. In particular, this makes it difficult to accomplish the goal of removing unwanted phrases, especially if they are present in a large number of hypotheses. An example of this issue is illustrated in the selected titles number 2 and 7

2. *PANASONIC PT-AE3000U Projector* ***Replacement Lamp with Housing***

7. *PANASONIC Projector* ***Replacement Lamp with Housing***

Phrases like *Replacement Lamp with Housing* occur in more than one title and would most likely receive a high arc score within the confusion networks, even though they should be penalized. However, approaches as such have no effective solution to penalizing these tokens only using information coming from the source title set.

1.4.2 Phrase-based Approaches

The phrase-based approach utilizes the re-decoding methods which involves collecting new phrase translation information for generating new hypotheses. This allows some re-ordering capability for the new model, but still lacks the long dependency information for long range re-ordering.

The primary goal of the phrase-based approach is to obtain sentence coherence by considering more than one token at a time. Some approaches adopt the hierarchical paraphrasing method [60], while there is also the lattice decoding approach which allows multiple phrases to connect to multiple other phrases, before the final decoding process that searches for the best paths (a major shift from the original CN approach).

The general process works as such: first extract the backbone (hypothesis that best represents all the others), then perform token alignment before extracting phrases using methods like TER-Plus (TER_p) [21]. Then form new phrase-associated edges in the network and finally select the best path in the lattice. As was mentioned in[59], it lacks the ability to re-order tokens within the backbone which becomes a caveat for the approach. The approach also assumes a general consensus among candidate hypotheses that will allow it to find the most representative backbone.

Moreover, definition of phrases in titles is something that warrants our research. Intuitively, a title is in its entirety a phrase. A title does not necessarily make a statement; nor does it need to have the usual linguistic components such as *subject*, *verb*, or an *object*. In fact, the above source title sets show that it needs not to even contain a *verb*. Furthermore, a title could be a sequence of *nouns* and does not need to abide by any grammatical rule. We can thus say that it falls outside of the scope of the typical linguistics realm. This complicates the nature of the task since the usual system combination techniques that utilizes language parsers [60] will not work as well as expected.

1.4.3 Sentence-based Approaches

Sentence-level models have more to work with from the beginning, since they consider the entire sequence. We could now extract information at multiple levels including both the token-level and phrase-level, and embed them into a traditional log-linear model [59, 60]. The approach is concentrated mainly on selecting the best hypothesis (or title) from k-best set.

This idea coincides well with the problem of title generation because the lack of sub-phrase units means that both the token and phrase levels alone are not suitable − each title is itself merely a phrase. To decompose titles into its sub-atomic units is not linguistically sound, since few tokens could conglomerate without having most of the tokens *glued* to be one single sentence phrase

Reference: *Panasonic PT-AE3000U LCD Projector*

Notice that other than the token *Projector*, any other tokens could hardly be a phrase on its own.

Other than the most relevant approaches, we want to also review some of the existing perspectives that also shed light on the task. Many of these problems are not exactly relevant, but do carry some measures of similarity that is worth mentioning.

1.5 Related Perspectives

In this section, we will first discuss the idea of summarization, then we will briefly touch upon both text generation and sentence simplification.

1.5.1 Summarization

Automatic summarization is generally categorized into two different types − abstractive summarization and extractive summarization. Abstractive summaries are formed by re-writing a shorter version of the original text with mainly phrases and words belonging to the author. On the other hand, extractive summaries are primarily a selection process where sentences from the original text are selected the way they are to form a new shortened version of the text [65].

On some levels, the task of Title Generation (TG) is analogous to abstractive summarization. Phrases and tokens from the source titles are re-used to construct a new title that most represent all of the titles. Instead, extractive summarization carries slight differences since it does not possess the decomposition and re-generation step, but it can be useful as part of the TG. It could serve as a component within the various approaches of system combination. Extraction summarization could be used to select the best title from the source title set taking into account both semantics and syntactics of titles.

1.6 Thesis Contributions Overview

We propose a novel approach for TG which comes in the form of a Generative-Discriminative framework that transforms the generation search space. The framework contains two major components. First, we have a feature-rich decoder that obtains semantic information from the source titles and then generates a new list of candidate titles. Then, we propose an attention-based neural re-scorer that allows us to point-wisely re-rank the decoded output set.

Our contributions include the following:

1. A Feature-rich Decoder (Word-level)

2. An Attention-based Neural Re-scorer (Sentence-level)

Feature-rich Decoder Instead of fusing tokens or phrases, we propose a sentence-level re-decoder (which we call decoder for short). It is analogous to the idea of re-decoder in the context of system combination. While the re-decoder constructs new phrase translation tables from the alignment, our decoder first collects token and source title set level statistics and then utilizes this information as a way to know the transition probability. This helps to prune the search space which could be exponential as size of source title set grows. The decoder is expanded using the idea of stack decoder with beam search using the threshold pruning. We found this to be an effective approach that allows us to achieve the primary purposes of , generalizability, robustness, and being able to function competitively in multi-lingual setting.

Neural Re-scorer We also propose an attention-based Recurrent neural network model that first encodes each states in the process hypothesis expansion. The encoding mechanism utilizes the bi-lingual token embedding which we trained with a sub-word model [7]. This allows the model to utilize distributional representations for rare words in an unsupervised

manner. These token embeddings account for the corpus-level information and help in the token selection process.

The re-scorer aims to account for long-term dependency within each title by: 1) constructing a richer representation that is the concatenation of both token embeddings and the sequence state features for each sequence state, 2) considering global features. A bi-directional Recurrent Neural Network (Bi-RNN) is fed with the sequence state encoding as mentioned in 1), before a sentential representation is created with the attention mechanism, which serves to reward or penalize sequence encodings accordingly. Lastly, global features provide additional contextual information to further enhance the final representation by contextualizing on information outside of the sequence.

All in all, we propose a novel and robust way of generating titles based on source titles. Moreover, this approach also offers a possible new perspective to Machine Translation's system combination. This is our first step towards a framework that resembles Generative-Adversarial Networks (GANs) [32] where our decoder will serve as the generator and re-scorer as the discriminator. Both models will then be trained jointly and be optimized to reach the global optimum in a zero-sum game.

Chapter 2

RELATED WORK

In this section, we will briefly touch upon the different existing perspectives on the Title Generation (TG). we will begin with the most relevant approaches, then expand the section in the order of extents of TG-specific usefulness.

2.1 Background

As far as we know, Title Generation (TG) is a unique task that does not exactly fall into the blanket of any existing Natural Language Processing (NLP) task. Prior works include the use of a phrase-based statistical machine translation engine to generate titles based on a rule-based system [62]. This work serves the same intention of generating titles but was given attributes of product which come in the form of slot-value pairs rather than source titles.

TG entails maximizing the conditional probability of a *sentence (or sequence of tokens)* given a *list of source sentences*. The generated target title is purported to represent the product pertaining to the list of source titles. Key information about the product might be absent from the inputs and additional information might be present in the source titles. For this reason, the task differs from common NLP tasks such as summarization, sentence simplification, Machine Translation (MT), and the System Combination. For instance, Neural Machine Translation (NMT) models are trained to maximize the conditional probability of *a translation sentence* given *another source sentence* over a large corpus [34]. However, TG assumes the absence of a large corpus and takes as input a varying set of sentences.

Owing to the novel nature of our task, several possible research directions could be pursued. Most related of which would be the system combination (or Multi-Engine Machine

Table 2.1: Table of Related Works

	Word	**Phrase**	**Sentence**
Hypothesis Selection	-	-	[85, 74, 40]
Re-decoding	[19, 30, 39, 68, 44, 37, 20], This work	-	-
Confusion Network Decoding	[27, 63, 76, 19, 38, 97]	[21, 24]	[76]

Translation). For convenience in this thesis, we will refer to it as system combination. The objective of system combination is to combine several translations from different systems in order to produce a better translation output. This is similar to TG as both system combination and TG involve the re-generation or selection of sentences (or token sequence) given the prior knowledge of a list of sentences (system combination task also includes the source sentence). Past literature have attempted to categorize approaches within system combination in a 2-dimensional way − 1) units of fusion, and 2) methods of combination. We have covered aspects in units of combination in the previous chapter.

Our focus now is to divide the system combination approaches into relevant categories. More recent works [59] have suggested dividing approaches into 1) hypothesis selection, 2) re-decoding, 3) confusion network decoding, 4) Lattice Decoding Model, and 5) Paraphrasing model. However, these perspectives do not necessarily apply to the task at hand. More specifically, titles can carry a certain amount of linguistic deviations from the typical English grammatical sentences in that they are generally missing a *subject*, *verb*, and an *object*. This makes both paraphrasing and lattice decoding models largely irrelevant for TG.

As such, we decided to go with the earlier categorizations [75]: 1) Hypothesis Selection [85, 74, 40], 2) Re-decoding [19, 30, 39, 68, 44, 37, 20], and 3) Confusion Network Decoding Models [27, 63, 76, 19, 38, 97, 21, 24, 76]. Moreover, we will consider 3 levels of units of combination which we have discussed previously: 1) word, 2) phrase, and 3) sentence. To get a better perspective, this general idea is shown in Table 2.1.

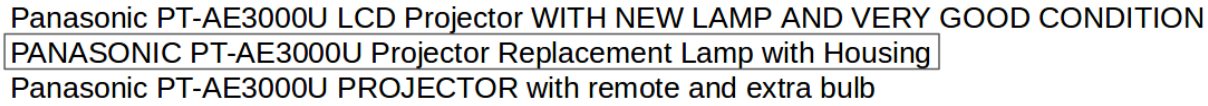


Figure 2.1: Selection Method

As previously mentioned, TG takes as input a source title set, and generates a title that not only preserves the essential attributes pertaining to the product, but also preserves the token ordering that loosely defines its grammatical structure.

TG involves construction of titles based on either the token-level or even the character-level conditional dependency path, as defined by the source title. The same intrinsic property exists in multiple tasks, including abstractive summarization, text generation, sentence simplification, and machine translation. We will also discuss these perspectives in the context of TG, using the following three titles as an example:

1. *Panasonic PT-AE3000U LCD Projector* ***WITH NEW LAMP AND VERY GOOD CONDITION***

2. *PANASONIC PT-AE3000U Projector* ***Replacement Lamp with Housing***

3. *Panasonic PT-AE3000U PROJECTOR* ***with remote and extra bulb***

We will begin with the most relevant context − system combination selection, then move on to discuss some of the existing ideas in other areas. We shall review all of sequence selection, re-decoding, and CN decoding in this order and then position our work more accurately as a re-decoding approach with units of combination at the word-level. This would further influence our decisions for approach comparison.

2.2 Hypothesis Selection

Hypothesis selection takes in many forms depending on the source of information. The fundamental idea for selection is to select a hypothesis that is not just the most representative

of all other candidates, but also the closest to the reference. In many scenarios, the hypothesis that possesses the greatest similarity through the means of BLEU [71], TER [21], or other metrics, might not be the one that is closest to the reference. This is a recurring issue in many of the approaches [21, 76, 27, 19, 85] since selection is an underlying aspect of all methods of combination. That is to say, we would anticipate the use of similar concepts in almost all systems.

Confusion Network Decoding involves selection of hypothesis as the backbone before sequence fusion. Errors in this step would ultimately propagate to the alignment step, and subsequently, to the last decoding step. In re-decoding approaches, partial sequences scoring and ranking are analogous to the selection approach since models need to have this discriminative ability in deciding which partial sequences should be kept during sequence expansion.

2.2.1 System Combination by Selection

Selective approaches are limited to the given hypotheses, and are thus prone to under-perform given a poor quality source title set. In the post-editing domain of Machine Translation (MT), We often need to know the translation quality before deciding if post-editing has to be performed [85]. This is the task of quality evaluation. Moreover, often this has to be done without the reference, which excludes the possibility of utilizing conventional metrics such as BLEU [71] and METEOR [3]. This specific scenario falls within the scope of this thesis which assumes no reference at test time.

In fact, approaches that merely select from the N-best set without refining the hypothesis search space rarely exist in system combination. Further, both re-decoding approaches and confusion network approaches are greatly interlinked with hypothesis selection as the last component of the combination. This idea is illustrated in Figure 2.2.

In most of the re-decoding approaches, a form of sentence-level quality evaluation is usually present. Trait-based approaches [19] consist of both decoding forests and a confusion networks which are generated for system combination, and the results show that there was no

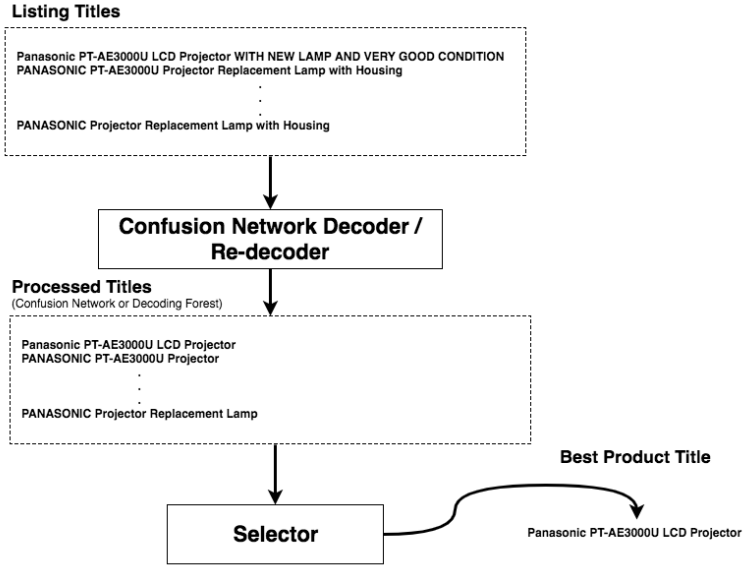


Figure 2.2: General Selection Pipeline

gain in BLEU scores moving from decoding forests to confusion networks. In fact, the work shows that decoding forest performs slightly better than the result of confusion network combination. It indicates that the hypothesis selection process is significant even when different system combination approaches are adopted, i.e. re-decoding and confusion network combination. This shows that it is important to have a high quality selector to discriminate between the diverse pool of candidate hypotheses.

Combination methods often produce ungrammatical token sequences, and raised the need for hypothesis confidence computation. In system combination, this selection is often achieved with the inclusion of a log-linear model with numerous features [76, 21, 27, 24, 30, 39, 19]. The model allows for inclusion of task-specific features such as Language Modeling (LM) probability, among many others. The objective is to model a long-dependency, grammatical sentence [76].

One benefit of sentence-level selection is that it can now avail itself of the overall semantic and syntactic knowledge sentence-wide. This gives room for the flexibility to design models

[NP Panasonic PT-AE3000U LCD Projector] [PP WITH] [NP NEW LAMP AND VERY GOOD CONDITION] .
[NP PANASONIC PT-AE3000U Projector Replacement Lamp] [PP with] [NP Housing] .
[NP Panasonic PT-AE3000U PROJECTOR] [PP with] [NP remote and extra bulb] .
[NP Quality Compatible Panasonic PT-AE3000U 165 Watt 2000-Hrs HS] .
[NP Panasonic PT-AE3000U PROJECTOR] , [NP remote,extra bulb] , and [NP INVIDEO HDMI] [ADJP switcher] .
[NP PT-AE3000U LCD 1080P Proj 60K] :[NP 1 1600 Lumens HDmi/component] .
[NP PANASONIC Projector Replacement Lamp] [PP with] [NP Housing] .

Figure 2.3: Shallow Parsing of Titles with UIUC Shallow Parser [73]

that look at the "traits" of hypotheses as if they were the final outputs. One example of this is the trait-based hypothesis selection that also utilizes the output-level features like "average output length" [19]. In most of these approaches, the goal is to minimize the distance between the 1-best hypothesis, or title in the context of this thesis, and the reference. Given a set of weights for f features $\theta = \{\lambda_1, \lambda_2, ..., \lambda_f\}$, reference title t_g, and metric function M, this can be written as

$$t_p = \arg\max_{\theta} M(t_p, t_g) \tag{2.1}$$

This optimization process is often done with the Minimum Bayes Risk (MBR) [52], which consists of a loss function based on the metric used. MBR decoding also incorporates syntactic structures and uses information such as word strings, word-to-word alignments, and parse trees of both source and target sentences. As such, MBR is relevant in a natural language setting where sentences can be parsed and analyzed in a grammatical setting. However, this would not work well in TG since parsing in titles is hardly effective as shown in Figure 2.3, where long, non-equivalent phrases are treated as a NP, making them difficult to be compared or aligned.

Multiple reasons render the traditional Natural Language Processing (NLP) to be futile in TG. One of which is the inconsistencies in token expression. Notice in 2.3 various forms for the token "projector" exist. This includes $PROJECTOR$, $Projector$ and $Proj$. As

such, there is a huge challenge awaiting any NLP methods in handling these titles.

Essentially, this argues for a more distributional approach that is able to represent tokens accurately by mapping multiple token's distributional semantics into a single vectorized form as what our approach has employed. We will next discuss about the extractive summarization methods in selection.

2.2.2 Extractive Summarization

Extractive summarization focuses on selecting salient sentences from an original document before combining them to form a summary. While this does involve reordering at the sentence level, the ordering of tokens is preserved.

A diversity-based ranking technique, Maximal Marginal Relevance (MMR) is often used as a simple and efficient way to prune the list of sentences [8, 28]. In many of the approaches attempted in the past, MMR is used as a way to reduce redundancy while selecting relevant passages for text summarization. While MMR is often used for diversity ranking in document retrieval, the approach demonstrated a clear advantage in removing unwanted sentences.

Many of these extractive summarization methods fall into three main categories: 1) methods simply based on sentence position or structure information, 2) methods based on unsupervised sentence ranking, and 3) methods based on supervised sentence classification [10].

Since the natural ordering of the source titles is not relevant to the summary construction, these approaches often pursue categories 2 or 3. The problem can be modeled as a regression problem with the label set as the n-gram overlap between the references and the summary [28]. More accurately, the problem can be modeled as short text summarization. This, for instance, has been done on the comment stream of a particular message from social network services. Visitors of social media sites intend to get the brief understanding of a comment stream without having to read the entire comment list. Similarly, short text summarization also provides a context in which the sentences are short and do not contain sufficient structural information [16].

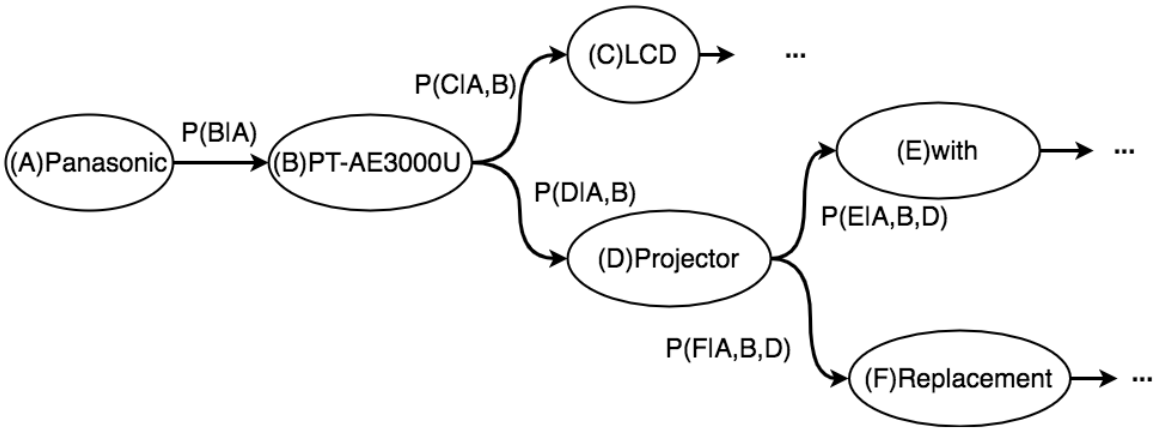


Figure 2.4: Re-decoding

With the popularity of neural networks, neural approaches have also been applied to text summarization. Early approaches include modeling the problem as a classification problem where sentences are represented as feature vectors, before being passed into 3-layer neural network to extract the hidden features for sentence saliency learning [45]. While this early approach has had some success, it does not have the component of generation, and is constrained to the existing source title.

2.3 Re-decoding

2.3.1 System Combination by Re-decoding

Re-decoding, or sub-sequence combination, is a combination method that generates and expands the hypothesis search space. This is done using features extracted from the partial sequences which translates to scores used in ranking of the partial sequences for pruning purposes. As shown in Figure 2.4, these scores come in the form of transition probabilities $P(w_t|w_0^{t-1})$. Most approaches adopted the Log-Linear Model where feature functions are taken as the weighted sum by multiplying the linear weight vector λ. Numerous works had been done in fine-graining these features. Some common features include length, Language

Modeling (LM) and the n-gram overlap [39]. One perspective on feature is that they serve the purpose of alignment. They could be loosely categorized into four types: exact matching, stem matching, synonym matching, and n-gram paraphrase [39].

The aim of re-decoding is to learn scores that most accurately represent the actual sequence quality score. Since scores are used to rank the partial sequences, accurate scoring function would yield the highest translation quality outputs.

One way to estimate the partial sequence scoring function is using the Minimum Bayes' Risk Combination (MBRSC), which combines outputs into a consensus translation with maximum expected BLEU score [31, 30]. In contrast with the sequence selection method, MBRSC could generate new consensus translations different from the original hypothesis set. Most of these approaches are BLEU-based, meaning that the idea is to minimize the BLEU risk [52, 31, 30].

Decoding algorithms range from dynamic programming (DP) paradigm [4], to stack decoding where k-best sub-sequences are maintained at each step of expansion. Search space can be represented as a directed acyclic graph where the states denote sub-sequences, and the edges are the tokens to be added [30]. Algorithmic expansions are focused on retaining high BLEU-scoring partial hypotheses. Due to the exponential number of states, there is a need for pruning, and the best-scoring hypotheses are kept at a ratio [30, 76]. Eventually, hypothesis search space is expanded and new hypotheses are formed. At this point, we hope to have coherent hypotheses in the search space, and the last step would be to find the best hypothesis from N-best hypotheses.

Since the best hypothesis is determined based on the sequence confidence score, one underlying issue with this approach is the lack of consideration for sentence-wide long-term dependencies with the sequence scoring function. Stack decoding entails the idea of sequence states where each newly expanded sequence is seen as a state, but this memoryless scoring function does not include sub-sequence scores from previous sequence states due to Markov assumptions. This assumption largely undermines the ability of these models, which stipulate that next sequence state depends only on the current sequence state and independent of

previous history.

2.3.2 Abstractive Summarization

Abstractive Sentence Summarization re-generates shorter sentences from the original ones while attempting to preserving their meaning. The aim is to produce a condensed representation of an input text that captures the essence of the original sentences [77]. We can draw a parallel between this task and TG where multiple titles are used as the source of information to produce one single output title. While abstractive summarization takes in the same input as that of extractive summarization, it is possible for it to perform word re-ordering, paraphrasing, and even generate semantically identical phrases to replace the original sentences [78].

Recent state-of-the-art approaches include the use of an attention-based conditional recurrent neural network models [78] with added convolutional layers [15]. These approaches incorporate less linguistic information, and make no assumptions about the vocabulary of the generated summary [78]. The effectiveness of these end-to-end approaches is evident as they outperform machine translation systems trained on the same large-scale datasets, yielding large improvement over the best system in the DUC-2004 competition [78, 15].

The main objective of the task neural abstractive summarization is to model the conditional log-probability of a summary as [78]

$$logp(y|x;\theta) \approx \sum_{i=0}^{N-1} log\ p(y_{i+1}|x, y_c; \theta) \tag{2.2}$$

where:

y_i is the special start symbol $< S >$.

C is the size of context.

x is the input sentence.

The formulation assumes the Markov assumption on the span of the entire sentence length, and the goal is to model on the local conditional distribution, i.e. $p(y_{i+1}|x, y_c; \theta)$.

Both models [78, 15] consist of a language model for estimating the contextual probability of the next word, and utilize an attention-based contextual encoder that construct a representation based on the generation context [78, 15]. A simple beam-search decoder is used to maintain a set of potential hypotheses before the last extractive tuning step. This last extractive step is similar to the hypothesis selection of the system combination, and the standard log-linear model is used to tune the weights for the small set of features [78].

However, two main problems exist with these approaches. One is the fact that they are data-driven [78], which makes them unsuitable for real world scenarios with limited datasets. For training, [78] uses the annotated Gigaword data set [33] which contains around 9.5 million news articles from different domains collected over two decades, and amounting to 4 million title-article pairs [78]. This is in sharp contrast with our objective − to train on a tiny training set, and generalize to a large unseen test set.

Secondly, the high rare word counts in the corpus of around 80,000 sets of source titles remains one of the concerns especially for our given title corpus, which contains a large amount of non-repetitive tokens.

Lastly, abstractive summarization is vastly similar to TG the nature of task is rather different since TG takes an input of multiple sentences rather than one.

2.3.3 Sentence Simplification

One of the objectives of title generation (TG) is to process long titles while preserving the content meaning. Thus, another way of looking at the problem is through the lens of sentence-level factual simplification (SLFS) [1]. SLFS is part of the broader spectrum of text-to-text generation problems, which includes summarization, sentence compression, paraphrasing, and sentence fusion. Recent approaches have treated SLFS as a monolingual text-to-text task and draw the inspirations from Statistical Machine Translation (SMT) [1].

One approach to the task has been the use of margin-based discriminative learning algo-

rithm on a handcrafted list of features [1]. This specially made feature set is used to model the syntactic and dependency structures of the sentences. Sentence generation is performed via the stack decoding algorithm in order to ensure an efficient process.

More recently, a deep reinforcement learning-based encoder-decoder model was applied to the simplification problem [101]. In the context of TG, the model would explore the space of possible generated titles, in order to optimize the simplicity, fluency of the title, while preserving the meaning of the input [101]. However, we will explore the capabilities of encoder-decoder models as a benchmark in our problem.

2.3.4 Neural Machine Translation (NMT)

With the rise of the neural approaches, neural machine translation (NMT) models re-defines re-decoding and is made of an encoder and a decoder. The encoder extracts a fixed-length representation from an input sentence of variable length; and the decoder generates a correct translation from this representation based on the knowledge learned from previous sentence pairs. In relation to TG, there is a need to select the source sentence, and then "translate" it to the target sentence, or the title in TG.

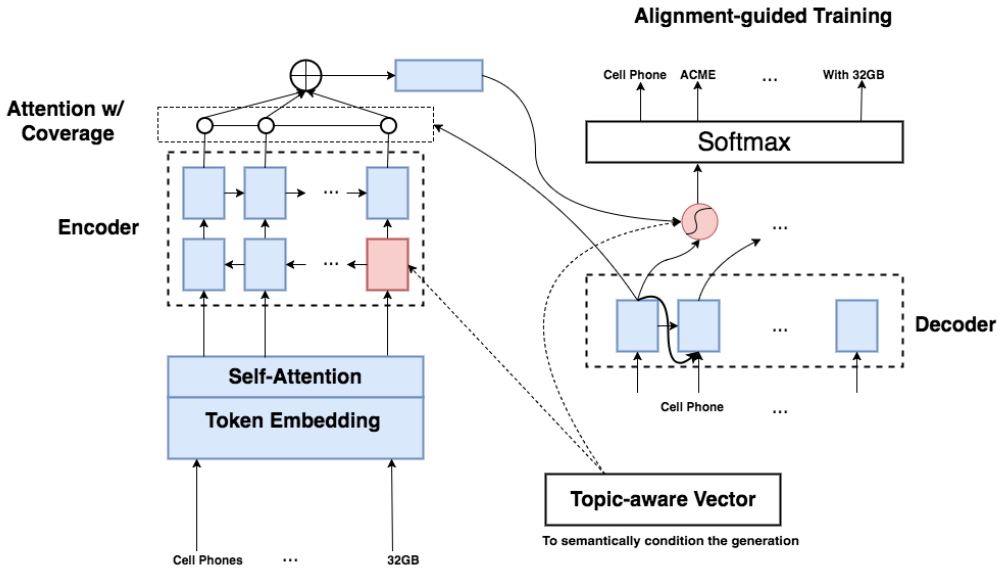


Figure 2.5: Conceptual encoder-decoder model with possible refinements [2, 11, 58, 72, 88, 91]

2.3.5 Encoder-decoder Model

We now discuss the various aspects of an encoder-decoder model used commonly in Neural Machine Translation (NMT) [2, 58] while also addressing some of the state-of-the-art refinements to the basic architecture.

Basic Architecture The encoder-decoder model, as the name implies, contains an encoder and a decoder. Encoder provide a representation for the entire input title, which consists of a sequence of tokens. Each token is fed into the encoder sequentially. Numerous works [2, 11, 35] use Bi-directional Recurrent Neural Networks (Bi-RNN) for the basis of encoders as shown in Figure 2.5. Output of encoder is the concatenation of outputs from both directions, $[\overrightarrow{h_k}; \overleftarrow{h_k}]$ at time k.

Decoder receives $[\overrightarrow{h_k}; \overleftarrow{h_k}]$ at all time steps k as input. Likewise, it consists of a recurrent neural network where at every time step k, the current hidden step would take as input the output from the previous hidden state, the last output word's embedding, and possibly

a context vector which is computed with attention mechanism from the encoder outputs. As shown in Figure 2.5, the last step of word prediction will have the decoder output pass through the softmax layer, which convert the prediction vector according to a probability distribution where the highest value dimension would correspond to the token prediction. As such, training process maximizes the word probability for the correct word at each time step with the negative log-likelihood function.

Conceptually, we see that each encoder state takes in information from 1) current token embedding, 2) last hidden state, which represents the entire sequence memory so far. On the other hand, each decoder state receives information from 1) entire encoder sequence memory up to the current time step, 2) entire decoder sequence memory up to the current time step, 3) last predicted target word output. Such is the holistic view of the dynamic between the encoder and the decoder. In terms of TG, the source titles will be encoded, and then decoded into the final title.

We now discuss some additional refinements that have been made to this basic architecture.

Attention Mechanism Attention involves computation of strength of association both between any two sources of information [2]. To account for the association between any input and other input, or self-attention, we take the dot-product between the encoded words. To account for the association between decoder states and the final encoder states, likewise, dot-products between them are computed before subsequent steps.

Several ways of computing the association strength have been proposed [58]. However, the subsequent steps to obtain the context vector remains largely similar. The raw association strength will then be normalized using the softmax function, and then any vectors fall within the context of consideration will then be weighted by the normalized association score. Therefore, the general steps to any form of attention is summarized as below:

1. **Compute association strength**.

2. **Normalize association strengths.**

3. **Scale the source vectors with the normalized association scores before taking their weighted average.**

Moreover, instead of computing the attention between input to output words, the idea of self-attention is also employed in several works [72, 89]. This helps to refine the representation of source tokens as contextual information is now part of the encoding. In self-attention, the association strength are calculated between words within a window [72], and step 2 and 3 remains largely similar.

This produces a final context vector. However, the encoder-decoder model still lacks the ability to take into consideration information that are not present in the original sequence of tokens. As such, several ways to account for this information were proposed.

Topic-awareness One way to account for the "topic", or in TG the source-specific information like category of the product, is to append to the existing Long short-term memory (LSTM) [42] cells an extra compartment in order to semantically condition the memory flow with external information [92]. This is shown in Figure 2.5 with the dotted line connecting topic-aware vector to the encoder's RNN (LSTM for this particular modification) cell. A more recent approach directly concatenate the topic-aware vector to the read-out layer before the final token prediction [11].

Guidance with Alignment While the domain information of the product could be encoded, it is desirable that fluency, or ordering of tokens could be enhanced. This is often done with alignment-guided training with modification to the cost function by including the alignment cost [**?**]. However this becomes a problem for TG, as we now need to select a source sentence that best represents the "source" sentence to be aligned with tools like GIZA++ [70].

In this work, we will restrict the bulk of our benchmark comparison with the focus on the most common version of the encoder-decoder model with attention mechanism, since it is a competitive architecture employed widely within the NMT community. We now discuss the bottleneck of the encoder-decoder models in general.

Model Challenges Numerous challenges were outlined in [49] including 1) domain mismatch, 2) amount of training data, 3) noisy data, 4) word alignment, and 5) beam search. In the context of our problem, NMT faces the issues related to domain mismatch, noisy data and amount of training data. Domain change often causes difference in machine translation system's performances. This is mainly due to the fact that tokens possess different meaning in different domain or contexts. As such, a system trained on some data points will most likely encounter a major non-overlapping vocabulary set, or high Out-Of-Vocabulary (OOV) counts, when being used to generate titles given source title set that are rather different. Our work is largely based on prior works in the Statistical Machine Translation (SMT) domain. Comparatively, NMT is on par with Statistical Machine Translation (SMT) in terms of in-domain performance, but performs poorly in its out-of-domain performance. [49] tests on five domains and found that the quality of system (BLEU) drops upon testing on different domains. In contrast to our task, we expect NMT systems to also suffer from this issue since each product could well be considered as an individual domain. As such, NMT will not be able to benefit from the sharing of word representation.

Besides, the given dataset contains a noisy data containing irrelevant phrases in high occurrence only within a certain source title set but are absent in any other product's source title set. In contrast, SMT is fairly robust to noisy data as it constructs probability distributions estimated from the corpus and noises influence only the tail end of the distribution. NMT yields poor performance with noisy data as its prediction has to reach a consensus between the language model and input context [49].

Moreover, although NMT is able to generalize better to larger context, it requires a huge amount of data to start performing well. In NMT, this often means training corpus sizes of

a few million words or less. This is undesirable in the industrial settings since it is simply too costly to obtain such amount of annotated data.

2.4 Confusion Network Decoding

2.4.1 System Combination by Confusion Network Decoding

A common approach to system combination is the Confusion Network (CN) decoding. In this particular approach, varying granularity of units are used to form the sentences. This includes both phrase-level [24, 21] and word-level [27, 76]. To construct a confusion network, there is a need to first select a backbone before word alignments with other hypotheses could be done. Backbone hypothesis has direct influence on confusion network-based approaches since it concerns the consensus translations directly [24]. Next there is the decision in choosing between 1-to-1 mappings [27, 76, 21] or the arbitrary, many-to-many mappings (lattice) [24, 60] in hypothesis alignment. The difference is that lattice is a more general form of confusion network [22], and thus it has a wider coverage and is expected to improve systems performance [24]. In relation to the task of TG, there is a direct correspondence between consensus translation and the best title. Thus, we will now review some of these approaches in greater details.

There is a variation of combination steps in CN decoding, although most approaches comfortably fall into the four-step strategy that is described above. A more subtle step is described in [24] where alignment pairs are normalized to ensure that the word order of hypotheses is consistent with that of the backbone. We will now describe in greater details the phases in CN decoding

1. Backbone Selection

2. Hypothesis Alignment

3. Confusion Network Construction

4. Decoding

Backbone Selection As previously mentioned, backbone selection is analogous in essence to sequence selection. Selected backbone will define the final word order of the system combination output [75]. One common technique is via the means of Minimum Bayes' Risk (MBR) decoding which uses BLEU as loss function. The model consists of a scaling factor which indicates the "quality" of each hypothesis with respect to the others. This factor is conceptually the same as the relative importance of each system in the Bayes gain computation [29]. In Figure 2.6, we demonstrate the selection of the backbone title from among three titles.

Suppose we have a total of N_s systems, and hypothesis t_b as the backbone candidate. Using TER_P as the loss function in MBR, the loss function would then be:

$$t_b = \arg\min_{t \in t_i} \sum_{j=1}^{N_s} TER_P(t_j, t_i) \tag{2.3}$$

There are two main ways to increase the robustness of the MBR decoder [21]. We will describe them in the context of TG. The first one is via filtering of worse or closely relevant individual titles based on metric scores, and keeping titles with similar quality [61]. The second way is to optimize directly on the development set with metrics like TER_P [21] or BLEU, before applying to the test set. The more successful mean is [21], and this approach is similarly adopted by the proposed model in this thesis.

Some approaches allow all hypotheses to play the role of the backbone once, creating one confusion network for each hypothesis. This approach becomes increasingly expensive as the number of hypotheses grows. Thus it was also suggested that bi-gram may first be used to expand the lattice [76], and then the weights for the expansion forest (confusion network in this case) are optimized. The reason why lower order n-grams such as bi-gram is considered is to avoid the heavy pruning. The process is then continued with a second set of weights for N-best list of the confusion network which is used to re-score hypotheses. This is done

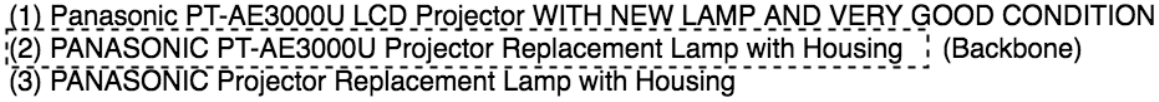


Figure 2.6: Backbone Selection

with higher order n-grams so as to find the best quality consensus output.

This thesis adopted similar procedure, but within the re-decoding domain of approach since no confusion network was constructed. Similar to [76], bi-gram is first used for expansion of the decoding forest, and subsequence quality are estimated with a set of neural weights. A second model is then used to determine the overall coherence of N-best list of candidate hypotheses. Figure 2.6 shows selection of a backbone sentence from a source title set of size 3.

Hypothesis Alignment Hypothesis alignment is the process in which hypotheses are aligned against the previously selected backbone. Two very different approaches have been proposed for alignment − pairwise and incremental alignment [75]. These ideas are clearly described in [75] as the following. Pairwise alignment aligns each hypothesis independently; incremental alignment initializes the confusion network by forming one word per link from the backbone hypothesis. The remaining hypotheses are aligned with the partial confusion network that allows words from previous hypotheses to be considered as matches. Thus in incremental alignment, the order of input hypotheses affects the alignment quality.

As such, five common alignment metrics includes 1) GIZA++ [70] 2) IHMM [38] 3) Inversion Transduction Grammars (ITG) [46] 4) Translation Edit Rate (TER) [81] and 5) Translation Edit Rate Plus (TER_P) [21].

More linguistic approaches include synonym matching that is proven to be beneficial to alignment accuracy. In fact, many more linguistic features are proposed to incorporate this prior knowledge for synonym matching [24].

(1) Panasonic	PT-AE3000U LCD	Projector	NULL	WITH NEW LAMP	NULL	NULL	AND VERY GOOD CONDITION
(2) PANASONIC	PT-AE3000U	Projector	Replacement	Lamp	with	Housing	NULL
(3) PANASONIC	NULL	Projector	Replacement	Lamp	with	Housing	NULL

Figure 2.7: Alignment

An example of hypothesis alignment is shown in Figure 2.7. In this example, both "PT-AE3000U" and "LCD" are aligned with themselves. However, "Projector" and "Proj", which are variations of the same token, are likewise aligned. For alignment considering synonyms, "LAMP" and "Lumens" will be aligned. Thus, Figure 2.7 shows the ideal alignment produced by a perfect aligner. Tokens in the rest of the sequence are dissimilar to each other both semantically and syntactically, and so they will not be considered.

Confusion Network Construction Variation for Confusion Network (CN) structures do exist. For the scope of this thesis, we will focus our attention on word-level network, which is essentially a directed acyclic graph which includes weighted edges/arcs and nodes. Each edge in CN denotes a token, and at times *null* token, and also the associated posterior probability [21].

On the other hand, lattice-based network's many-to-many mappings align phrase pairs, thus reducing the potential ungrammaticality, increasing the transitional coherence [24]. This idea is illustrated in Figure 2.8, where empty strings are denoted with "$". Note that in this graph, less than half of the backbone sequence (only 4 out of 11 tokens) are aligned for the backbone. Such is the recurring phenomenon among source titles, and poses a great challenge to titles that contain misspelled tokens. The difficult part would be for the system to know for example "projector" and "proj" are essentially the different spelling of the identical token. Though String measures like Levenshtein similarity [53] could map them to the same token, still this does not resolve the issue of word-level semantic similarity, which is crucial when string similarity measures become erroneous, and there the need for semantic similarity arises.

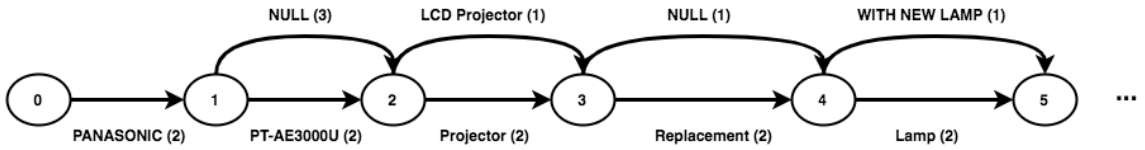


Figure 2.8: Confusion Network (CN)

One major problem with CNs is that they depend heavily on the selection of the backbone. This is a challenge when erroneous backbone is selected, and the error will propagate to the decoding step. Another issue is that hypotheses might have very different orders, complicating the alignment process [75]. On the other hand, some approaches make each of M hypothesis to be the backbone, forming M confusion networks [63]. This proposal essentially leaves the decision to select the backbone to decoding time by combining networks built using each output as a skeleton into a large lattice [75]. This not only means that we will now consider all possible paths, we also would have a computationally expensive process when the number of titles grows to be huge (e.g. in one source title set the number of titles is 7,500). As such, it is not ideal in the task at hand.

Decoding CN consists of multiple hypothesis paths. The last step is to select the best consensus hypothesis within the expanded search space. Log-linear combination with various feature is used for decoding [76]. Just like in hypothesis selection, log-linear model is the common approach to decoding and selection of the best hypothesis to be the consensus translation. The main objective of this is to estimate the token posterior probability based on weighted feature functions. Since this step is similar to the re-decoding step, we will not go in detail.

While Confusion Network Decoding provides a mean of generation, it is not without disadvantages. Particularly, the selection of the backbone sentence and subsequently the confusion network construction is an expensive process when the number of titles increases

to a large amount. This excludes the consideration for the more advanced approaches where each title in the source title set is treated as a backbone sentence once, thus constructing an exponential number of confusion networks.

2.4.2 ASR Summarization

In the context of ASR, there often exist a high speech recognition error, e.g. on meeting recordings. As such, an unsupervised approach using confusion networks could be used to improve summarization performance [96]. Similarly, source titles often contain unwanted tokens or titles as noises that are to be weeded out. Confusion Networks (CN) is a more compact and powerful representation for word candidates. Similar to Confusion Network Decoding, ASR summarization involves the construction of confusion network based on the source title set before the final decoding step. As such, this approach encounters a computational bottleneck when given a large source title set.

2.5 Relevant Perspectives

Now, we will discuss two relevant perspectives with regard to all the approaches mentioned above. This consists of 1) log-linearity and log-non-linearity and 2) generative and discriminative models.

Log-linear and Log-non-linear Models Decoding of approaches mentioned previously utilizes the the log-linear model for partial sequence score estimations. However, the model is not without its caveats. We now discuss some of the underlying assumptions for the use of log-linear model.

Log-Linear Models are also known as the Poisson regression, which is a generalized linear model in regression analysis. However, Poisson regression is optimal when the mean of the data distribution and variance are identical. In other words, it assumes the Poisson distribution to have limitless number of independent realizations of a Poisson random variable

[5]. It also follows that for real data generated by a Poisson process, the mean should be approximately the same as the variance.

However, this is not the case for a limited dataset, which often does not satisfy Poisson distribution. In fact, many of the decoding features contains observed feature data where variance is greater than the mean. This is an indicator that the log-linear model is not fitting to be used for this task, where data is scarce. We will further perform an analysis for feature data in the last chapter to show that variance of data is indeed greater than the mean for all features.

Here our main concern is the suitability of log-linear model for estimating the sub-sequence confidence during hypothesis expansion, and also hypothesis selection during the re-scoring phase. We will now discuss the theoretical reasons to move from linearity to non-linearity.

Neural Network (NN) is a flexible nonlinear regression model, and can always be used to approximate a continuous function. It is effective for response variables or target values that are real-valued [23]. It has also been shown that nonlinear extension of Poisson regression based on neural networks not only has the capability to learn linear relationship, but also to model the non-linear relationships between features [23]. As such, neural models are better able to generalize to large unseen data since more complex equations can be modeled.

In other words, the linearity assumption in Poisson regression (or log-linear model) is disadvantageous to modeling our estimators. In practical situations, features do interact non-linearly in the high dimensional feature space and for this reason we opt for a neural re-scorer, instead of the traditional log-linear model approach for sequence confidence estimation.

Empirical studies have also been done in shifting from linear to non-linear modeling when dealing with distributional feature representations in a continuous space [90]. In fact, it has also been shown that in the domain of Natural Language Processing (NLP) non-linear models are highly effective in low-dimensional distributional spaces [90]. This empirical study was done in the task of sequence labeling (specifically Named Entity Recognition and Syntactic Chunking). Furthermore, the study also showed that distributional representation could be

used to generalize effectively to large unseen data [90], which is the intention of our work.

Generative and Discriminative Models Most of the above-mentioned approaches have components of either generative or discriminative nature. Re-decoding approaches adopts a discriminative log-linear model (a form of linear regression) that allows it to generate token sequences. We will now discuss our proposed solution through the lens of generative and discriminative models.

Generative classifiers learn a model of joint probability $p(x, y)$ of, in our case, source titles x and the reference title, y. A trained generative classifier then selects the most likely y probabilistically via computation of $p(y|x)$. On the other hand, discriminative classifiers model the posterior $p(y|x)$ directly, learning to map from source titles x to class labels, or a score indicative of similarity of x with respect to y [67].

Generative Adversarial Networks (GANs) achieved recent successes in various fields [32]. It is a unifying framework that combines both generative and discriminative models. The neural model consists of a generator G and a discriminator D and is optimized by 1) maximizing the probability of assigning the correct label (produced by D), or score, to both training data and samples from G. Simultaneously, the framework consists of an algorithm that 2) minimizes the loss of generation by G. Such framework is seen in Neural Machine Translation (NMT) [93] where encoder-decoder output serves as the generator and an additional discriminator, made up of Convolutional Neural Networks (CNN) [51], is proposed.

In relevance to our work, our proposed solution consists of both a decoder that serves as a generator and a re-scorer that acts as the discriminator. Though they are not not jointly optimized as in GANs in an adversarial manner, but are optimized in a sequential manner more akin to boosting [25].

2.6 Summary

In this chapter we provide an overview of related work pertaining to Title Generation (TG). We discuss Natural Language Processing (NLP) domains along the dimensions of 1) hypoth-

esis selection, re-decoding, and confusion network decoding. Hypothesis selection approaches include system combination by hypothesis selection, extractive combination. Re-decoding approaches include system combination by re-decoding, abstractive summarization, neural machine translation and sentence simplification. Lastly, confusion network decoding is particularly used for system combination, but also appears in Automatic Speech Recognition (ASR) summarization.

Our work falls into both the category of hypothesis selection and the re-decoding. In this thesis, we compare our work against approaches in both the hypothesis selection and re-decoding blankets. In the next chapter we will discuss in detail the methodology involved and the theoretical reasonings for our proposed framework.

Chapter 3

METHODOLOGY

3.1 Overview

In this chapter we will discuss about the details of how our proposed framework of Title Generation (TG) works. We will start off by defining standard of measurement for the quality of a title. Then we will formally introduce our Generative-Discriminative framework which consists of a decoder and a re-scorer.

3.1.1 Evaluation Metric

Essentially, quality of a generated title is proportional to the similarity it carries with the target title. This idea of similarity can be measured with respect to the number of overlapping tokens between them. One of such automatic metric used commonly in the machine translation is BLEU. BLEU can be used to measure the closeness of the generated title to the reference title. It measures the closeness via computing, in our case, the sentential overlap between generated title and reference title. Ideally, a computationally inexpensive metric is desired to suit the needs of the industry. BLEU as an automatic metric that requires no training could quickly determine the quality of our candidate title T in a highly correlated manner with respect to the actual human evaluation [71]. Mathematically, BLEU is a function that considers n-gram overlap, lengths of both candidate T and reference R. In this thesis we consider the sentence-level BLEU [9]:

$$BLEU(N, T, R) = P(N, T, R) \times BP(T, R) \tag{3.1}$$

where N (n-gram size) in our experiments are assumed to be 2. $P(N, T, R)$ is the geometric mean of n-gram precisions:

$$P(N,T,R) = (\prod_{n=1}^{N} p_n)^{(1/N)} \tag{3.2}$$

where:

m_n is the number of n-gram overlap between system candidate title and the reference title.

l_n is the total number of n-grams in the candidate title.

p_n is given as $p_n = (m_n)/(l_n)$.

A multiplicative brevity penalty factor is introduced in order to avoid the risk of having very short titles that have maximum precision but are very far from the actual complete title. It is defined as according to the following equation:

$$BP(T,R) = min(1.0, exp(1 - len(R)/len(T))) \tag{3.3}$$

We also adopted smoothing method 4 described in [9] which assigns proportionally smaller smoothed counts for shorter candidate titles, due to the concerns for inflated precision.

3.1.2 Definitions

Given the evaluation metric, we will now formally define the best title t_p with the following definition:

Definition 3.1.1 *The best title t_p for source title set T is a token sequence within a search space S that has the highest sentence BLEU relative to the reference title t_g. This is given as*

$$t_p = \arg\max_{t \in S} BLEU(t, t_g) \tag{3.4}$$

Note that this definition also implies that at test time, we will need a way of knowing the quality of any given title. As will be introduced in later sections, this is accomplished by training a function that estimates the sentence BLEU of a title given information about the source title set and the title.

Framework Definitions Before we proceed to our model, we will re-define some of the definitions stated in [6], and adapt them later on to our proposed framework.

To be clear, K represents the space of reference titles, Y indicates the best title output. Loss function computes the dissimilarity between K and Y which in our formulation is determined by the BLEU score.

Definition 3.1.2 *A supervised learning problem is a tuple (D, T), where $T = (K, Y, l)$ is a supervised learning task, X is an arbitrary feature space, and D is a distribution over $X \times K$.*

Definition 3.1.3 *A supervised learning algorithm for task (K, Y, l) is a procedure mapping any finite set of examples in $(X \times K)^*$ to a hypothesis $h : X \to Y$.*

Therefore, our framework will be a supervised learning algorithm for task (K, Y, l) in order to achieve the goal in solving a supervised learning problem is to find a hypothesis $h : X \to Y$ in order to minimize the expected loss as defined by BLEU. These definition will be used in Definition 3.1.4 as a way to describe the goal for framework components.

3.1.3 Generative-Discriminative Framework

Framework Motivation In our related work chapter we have discussed the three major approaches consisting of 1) hypothesis selection, 2) re-decoding, and 3) confusion network decoding (in order to avoid confusion with later addresses, we will use the word "sequence" instead of "hypothesis" in reference to the generated titles). We also noticed the overlap between all three approaches in that a form of sequence selection is used in all approaches. This motivates our work in this thesis − we propose to provide a unifying framework that

helps to reconcile sequence selection with the other two approaches. As such, we will now introduce the Generative-Discriminative framework consisting of a decoder (generative) and a re-scorer (discriminative).

The generative model works similarly in spirit to the ensemble meta-algorithm of boosting. In a boosting framework, several classifier are trained sequentially where a data distribution is iteratively discerned by classifiers. In our case, we are utilizing a decoder and a re-scorer as two different classifiers to make judgments regarding the quality of the title search space. Unlike boosting, this is an extreme scenario where some data points are assigned an importance score of zero (or discarded without replacement) by the decoder, before being passed to the re-scorer for the final selection.

Due to the expenses of computation, the translation process in Statistical Machine Translation (SMT) often has a decoder that is designed to be efficient and tractable. As such, more refined, feature-rich re-scorer is preceded by the decoder in order to select the best translation output from the k-best list. This provides the practical motivation for this framework.

Theoretically, for systems that output a large variance of data, it is often difficult to pinpoint the particular range of highest-quality titles. That is to say, our initial candidate titles in the search space constitute of a range of titles where the titles of concern lie mainly in one particular region − in the highest BLEU-scoring range of candidate titles. As such, the generative component of our framework helps to narrow down the space of search space by first reducing the search space (decoder) and then select from it (re-scorer). Specifically, by "generation" we meant that the BLEU variance of the initial source titles are reduced and limited to be lower. As mentioned, the consequence of that is a search space with better quality titles with which the re-scorer can select from.

One other major advantage that our Generative-Discriminative framework offers is its flexibility in allowing different approaches to be incorporated. For instance, the task of system combination could be studied by having a generative component and a discriminative component where sequence selection takes place. Progresses made in the generative component with confusion network decoding or re-decoding will result in progress for the

discriminative component of sequence selection and vice versa.

Framework Formulation We model the task as a supervised learning problem that contains (Input X,Label Y) pairs which corresponds to (Source title set T, Target title t_p) pairs in the context of Title Generation (TG). We can then formulate the Generative-Discriminative framework for the task with the following probabilistic decomposition:

$$\begin{aligned}
\Pr(X,Y) &= \Pr(Y|X) \times \Pr(X) \\
&= \Pr(t_p \approx t^*|T) \times \Pr(T) \\
&\approx \Pr(t_p \approx t^*, T^{'}|T) \times \Pr(T) \qquad (3.5) \\
&= \Pr(t_p \approx t^*|T^{'},T) \times \Pr(T^{'}|T) \times \Pr(T) \\
&= \Pr(t_p \approx t^*|T^{'},T) \times \Pr(T^{'},T) \\
&= \underbrace{SQE(t_p \approx t^*|T^{'},T)}_{\text{Discriminative}} \times \underbrace{STE(T^{'},T)}_{\text{Generative}} \qquad (3.6)
\end{aligned}$$

where:

t^* is the human-annotated target title that might possibly not be attainable given source title set, T.

t_p is the attainable target title given the source title set and that the framework aims to generate.

$T^{'}$ is the intermediate outputs from decoder.

$STE(\cdot)$ is the generative model that represents the first stage of our framework.

$SQE(\cdot)$ is the discriminative model that helps to distinguish the true distribution X as represented by the target title t_p.

Equation 3.5 is where the framework is divided into generative and discriminative models. Here we assume the presence of a generative model (decoder) output $T^{'}$ which is taken to be the best set of possible outputs given source title set T. The main reason for such assumption is the fact that not all target titles can be generated given the source title sets. Tokens appearing in target title might not exist in the source title set, or that the language modeling of target title is not attainable within the source titles. All these reasons would render our task a difficult one and we will instead aim to attain the best title t_p given the source title set T.

We reformulate the joint probability of both the decoder output set $T^{'}$ and target title set to be the product of the probability of target title conditioned on both distribution of decoder output and $T^{'}$ and probability of decoder output conditioned on source title set T. Equation 7.1 represents the backbone of our Generative-Discriminative framework. It indicates the two major components of our framework − a generative decoder and a discriminative re-scorer.

Initial Search Space: Bi-gram Forest Before proceeding to the framework, we will first discuss the title search space that serves as the pre-conditions to our generative decoder.

Recall equation 1.2:

$$t_p = \arg\max_{v \in V, d \in D} P(v^d | T, O) \tag{3.7}$$

The initial idea for the search space is that all the possible combinations of tokens are defined by the power set of vocabulary V with orderings defined by the set of all possible ordering set D. This potentially large search space would render the search process computationally expensive. Instead, the problem with token selection and token ordering can be resolved when we examine the fact that source title set T defines the natural token ordering via the basic idea of language modeling. Since t_p is purely generated based on T as the source of Language Modeling (LM), the perplexity of LM_T is 1. We know the perplexity of a discrete probability distribution p is defined as:

$$2^{H(P)} = 2^{-\sum_x p(x) \log_2 p(x)} \tag{3.8}$$

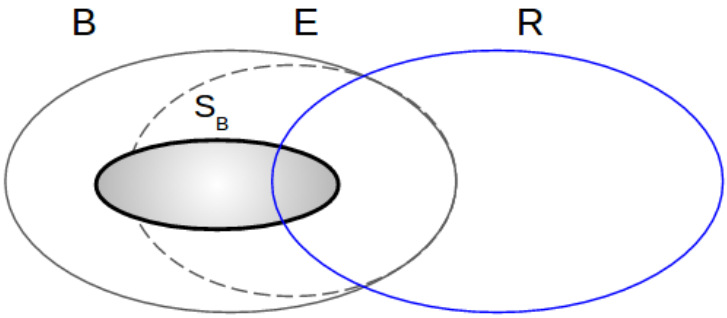


Figure 3.1: Initial Search Space: B is the original bi-gram forest as defined by source title set T, E is the pruned title space, and R is the space of target titles.

Since all the tokens used for generation are taken from the source title set T, $p(x)$ will approach 1, resulting in $-\sum_x p(x)\log_2 p(x)$ becoming 0 and $2^{H(P)}$ going to 1. This shows that the LM_t is capable of making good predictions. With that said, we propose to use the token ordering d to be determined by the LM_t derived from source title set T.

As such, our initial search space B is defined as

$$B = \{x_1^B, x_2^B, \cdots x_m^B\} \tag{3.9}$$

B is the set of token sequences $\{x_1^B, x_2^B, \cdots x_m^B\}$ as defined by the source title set's bi-gram probabilities. As shown in 3.1, B defines our feature space and S_B is the conceptual set of selected titles within B.

The original source title set T defines the search space B, which is the space of all possible bi-gram-based token expansion. The main goal of the generative component is to reduce B to E (the pruned search space) in order to reduce variance of the dataset by maintaining only the BLEU-correlated data distribution so that less samples are needed for the model to generalize. We will now describe the framework components.

Framework Components Our Generative-Discriminative framework consists of a generative decoder and a discriminative re-scorer. The generative decoder consists of an algorithm

that decodes by using a BLEU estimator which also serves as a state transition probability estimator. The decoding algorithm is based on Hidden Markov Modeling (HMM) which consists of 1) a number of states, and 2) topology of the states, i.e., how they are connected. The BLEU estimator is the State Transition Estimator (STE) that is trained on the original search space B. It prunes the entire bi-gram forest by maintaining only a portion of B based the the sequence's BLEU score with respect to the reference title. The result is a reduced and shifted data distribution in the pruned search space E. Assuming there are n titles in the search space E, then it is the set, where x_i represents the token at positino i:

$$E = \{x_1^E, x_2^E, \cdots x_n^E\} \tag{3.10}$$

The discriminative re-scorer is an RNN-based BLEU estimator, which we call the Sequence Quality Estimator (SQE). It learns to discriminate and select the best hypothesis h_E from the pruned search space E. It does the final job of selection the best sequence from E.

With B being the bi-gram search space, E as the pruned search space, and R as the reference title space, we define the high BLEU-scoring search space S_x as the set of titles selected by learned hypothesis h_x learned from space x. From Figure 3.2 we could learn the following: (i) $E \subset B$ and so $n < m$. This gives us two such search spaces, S_B and S_E. (ii) $(B \cap R) = (E \cap R)$ which means that the actual high BLEU-scoring region as defined by $B \cap R$ did not increase. We merely increase the likelihood that the new selected space S_E will cover a large portion of R. This then leads to the idea of variance reduction with the bi-gram search space pruning with the generative decoder, which we now describe.

Variance Reduction The idea of variance reduction stems loosely from the concept of sampling. Fundamentally, decoder's generation is a form of non-uniform sampling where higher BLEU-scoring titles are assigned a higher score and consequently chosen (by decoder) to be the subsequence distribution for the re-scorer. In essence, the framework chooses more samples that contribute more to the result (in our case high BLEU-scoring titles), and less

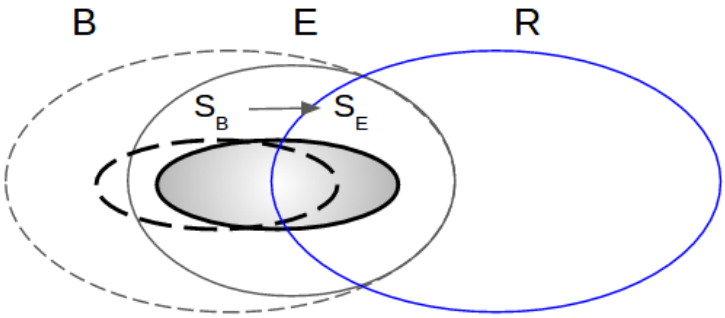


Figure 3.2: Transformed Search Space

samples that have minor contributions (low BLEU-scoring titles). This works by utilizing prior information learned from the initial distribution, and then make posterior predictions. We will now formalize the generative component of our Generative-Discriminative framework.

Formally, the main objective of our generative model is described below:

Definition 3.1.4 *A **generative reduction** $R(B, A)$ from problem (D, T) to problem (D', T) takes as input a finite set of examples $B \in (X \times K)*$ and a learning algorithm A for problem (D, T), and outputs a hypothesis $h : X \to Y$ that is used to sample a subset of the original title space B to form a pruned title space E in order to produce a **higher BLEU-scoring distribution**. Our generative model aims to achieve this:*

$$l(k_B, y_B) \geq l(k_E, y_E) \tag{3.11}$$

where $l(k_i, y_i)$ is the previously-defined loss function that computes the loss between prediction y_i and label k_i. Algorithm A would be the previously-mentioned decoder, and the resulting samples in E will have a reduced variance and experience a shift towards high BLEU-scoring region. Our eventual goal is to learn a hypothesis that is the most BLEU-correlated. Thus for any test example (k, y) and feature x, we intend to find the hypothesis h' such that

$$h' = \arg\min_{h \in H}(l(k, h(x) = y) \tag{3.12}$$

As such, we could reduce the variance and maintain only the estimated higher BLEU-scoring region, which saves the need to train on large data set for title selection. We postulate that this shift is necessary and the idea is supported by the empirical results.

We will now outline both the generative and discriminative models within the our framework. This consists of the Sequence Transition Estimator (STE) and the Sequence Quality Estimator (SQE). E is the external knowledge, T is the source title set, v^d is a generated title in the generated (or pruned) title set $T^{'}$.

Sequence Transition Estimator (STE) During the process of title sequence expansion, we wish to maintain a list of candidate titles as shown below following from equation 1.2:

$$\begin{aligned} t_p &= \arg\max_{v \in \mathcal{P}(V), d} P(v^d | T, E) \\ &\approx \arg\max_{v \in \mathcal{P}(V), d} \big(\prod_{t=1}^{|v|} P(w_t | w_1^{t-1}, T, E) \ \big) \\ &= \arg\max_{v \in \mathcal{P}(V), d} \big(\prod_{t=1}^{|v|} STE(w_1^t, T, E) \ \big) \end{aligned} \tag{3.13}$$

where $STE(\cdot)$ is a the parameterized decoder function that is to be optimized by minimizing the BLEU [71] between partial token sequence v^d and reference title t_g.

In order to maintain a k-best list of partial sequences so as to retain potentially better titles, we apply stack decoding and expand the sequences one token at a time. Thus, at time t during the hypothesis expansion of the decoding process, we will select a new set of tokens W_t to be appended to each of the hypotheses, where $\Phi(\cdot)$ denotes the function that return the history of a token

$$W_t = \bigcup_{i=1}^{k} \Big(\arg\max_{w_t^i \in V \ \setminus \ w_t^{i-1}} STE(w_t^i | \Phi(w_t^i), T, E) \Big) \tag{3.14}$$

Sequence Quality Estimator (SQE) To select the best title, we propose to utilize a sentence-wide re-scorer in order to rank the fully expanded hypotheses at the end of the decoding process. Suppose T' is the set of k-best generated titles, the basic idea is given as

$$\begin{aligned} t_p &= \arg\max_{t_h \in T', d} P(t_h | T, T^{'}, E) \\ &= \arg\max_{t_h \in T', d} SQE(t_h | T, T^{'}, E) \end{aligned} \tag{3.15}$$

where $SQE(\cdot)$ is the parameterized re-scorer function that takes as input a full hypothesis sequence and returns as output a BLEU estimation.

To summarize, our framework consists of two main components: 1) candidate title generation with the decoder and 2) re-scoring of k-best candidate titles in order to select the best title. The primary goal of the decoder is to generate a search space of reference-correlated candidate hypotheses. Given the limited data size, both decoder and re-scorer ought to be optimized on a limited set of training data, and generalize to large unseen data. Data points for the decoder come from individual partial sequences; while the re-scorer is trained on data that is sentence-level, so as to reduce the reliance on decoder's estimation.

Thus at this point, it is now clear that our framework is two-fold. we will need to model two estimator functions:

- Generative Sequence Transition Estimator (STE) for generating the set of candidate titles T', and

- Discriminative Sequence Quality Estimator (SQE) for evaluating candidate titles for selecting t_p

To conclude, we discuss the roles of Sequence Transition Estimator (STE) and Sequence Quality Estimator (SQE) with respect to the title generation process. Essentially, the initial set of source titles $T = \{t_1, t_2, ..., t_m\}$ will be passed as input to STE that then produces a generated list of titles $T' = \{t_1, t_2, ..., t_g\}$. Finally, SQE selects the best titles $t_p \in G$.

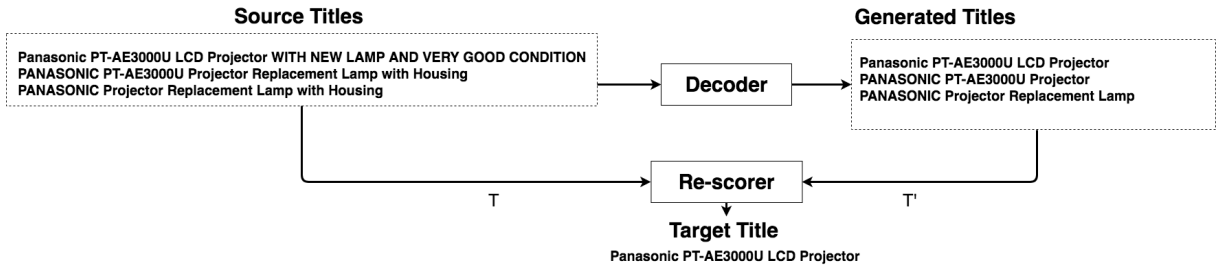


Figure 3.3: General Pipeline: Figure shows the transitions $T \rightarrow T' \rightarrow t_p$

Moreover, we pre-process the titles as a form of normalization. Source title sets are also filtered to see if the algorithmic speed is improved. We discuss these ideas in the next section.

3.2 Preprocessing

The idea of preprocessing is to eliminate the amount of noise present in the original source title set. This objective can be broken down into two components − 1) source titles filtering and 2) tokenization of the source titles. The latter would simply be the removal of punctuations, symbols, or the normalization of sentence contractions. In other words, sub-words that appear as separate tokens should be considered as one. For example, there are instances where 50 mm appears as 50mm, depending on the titles. Such combination reduces complexity in the problem and also helps to produce more accurate language modeling as it refines the token boundaries. We use string similarity measures to combine these tokens.

Our preliminary experiments found that the quality of the initial source title sets directly influences the performance. We also found that the quality of input titles prior to the decoding process helps in raising the performance. We intend to explore with the traditional Maximal Margin Relevance (MMR) approach while setting the threshold λ, or selection ratio, to be 0.5. This is equivalent to selecting the first 50% of all the source titles and therefore analogous to splitting the source title set into half while maintaining the token distribution according to token relevance in the process.

The idea of MMR is simple. Given source titles T, our goal is to select a subset T' of

titles from T. MMR builds T' in a greedy manner by selecting t_i from T. Selected title t_s is performed as the following

$$t_s = \arg\max_{t_s \in T \setminus T'} [\lambda Sim_1(t_s, c) - (1 - \lambda) \max_{t_i \in T'} (Sim_2(t_s, t_i))] \tag{3.16}$$

where $Sim_{[1,2]}$ is the similarity function which is defined as the cosine distance between the two sentence vectors where each word is represented by the Term Frequency Inverse Document Frequency (TF-IDF) [57, 83] scores. TF-IDF is calculated as

$$tfidf(t, d, D) = tf(t, d) * idf(t, D) \tag{3.17}$$

where:

t is the term.

d is the current document (title) where t resides.

D is the set of all documents.

While $tf(t, d)$ [57] simply denotes the number of times that t occurs in D, the formula for $idf(t, D)$ [83] is given as:

$$idf(t, D) = log(\frac{|D|}{|\{d \in D : t \in d\}|}) \tag{3.18}$$

Having discussed some pre-processing steps we attempted, we will now formally describe components within our Generative-Discriminative framework. We begin with the generative decoding, and then elaborate on the discriminative re-scoring.

3.3 Generative Decoding

The process of decoding is based on the Hidden Markov Modeling (HMM), which consists of 1) transition probabilities between states and 2) graphical structure and topology of states. 1) will be introduce in subsection 3.3.2 and 2) will be discussed in subsection 3.3.3.

3.3.1 Background

The process of decoding is analogous to the decoding process of Statistical Machine Translation (SMT). We will start by presenting the phrase-based translation model:

$$e_{best} = \arg\max_{e} \prod_{i=1}^{D} \phi(\overline{f_i}|\overline{e_i}) \times d(start_i - end_i - 1) \times p_{lm}(e)$$

where:

$\phi(\overline{f_i}|\overline{e_i})$ is the phrase translation table, which contains information of the probability distribution between phrases in the source language and phrases in the target language.

$d(start_i - end_i - 1)$ is the distance-based reordering model based on distance to end position of previous input phrase.

p_{lm} is the language model probability.

Title Generation (TG) does not require a distance-based re-ordering model since orders are determined by the Language Modeling (LM) given by the source title set T. In this generative model, decoding process is analogous to HMM where monolingual translation table statistics correspond to the emission probability while LM probability represents the transition probability.

In our formulation, we have the Sequence Transition Estimator (STE) which provides partial sequence scores during the generation process. At this point, we are only concerned with the token-based model, and we leave the phrase-based approach for future work. In light of the SMT formulation, we will now formulate the State Transition Estimator (STE).

3.3.2 Feature-rich Estimator $STE(\cdot)$

To begin, we will first introduce the idea of Log-Linear Model, and then relate it to our feature function, before finally discussing the features.

Log-linear Model The proposed $STE(\cdot)$ uses the idea of the feature functions $f(\cdot)$ of log-linear model. To approximate the function $p(\cdot)$ which takes as input x with a total of n features. Then $p(x)$ would be given by

$$p(x) = \exp(\ \sum_{i}^{n} \big(\lambda_i\ f_i(x)\big)\ ) \tag{3.19}$$

where

f_i is a feature function derived from information within the source title T.

As such, we start with the translation model, and modify it to be a monolingual translation model. We let the current best generated title during the generation process be t_g, which is the partial token sequence. Thus t_g would then be given by:

$$\begin{aligned} t_g &= \arg\max_{v \in V, d} P(v^d|T, E) \\ &\approx \arg\max_{t \in T'} \prod_{i=1}^{|t|} \phi(\overline{e_i^t}) \times d^t(start_i - end_i - 1) \times p(\overline{e_i^t}) \\ &\approx \arg\max_{t \in T'} \big(\lambda_1\ l(w_1^t \cdots w_{|t|}^t|C_t) + \lambda_2\ s(w_1^t \cdots w_{|t|}^t)\big) \\ &\approx \arg\max_{t \in T'} \big(STE(\hat{l}_t, \hat{s}_t)\big) \end{aligned} \tag{3.20}$$

where:

$\phi(\cdot)$ is the phrase probability of $\overline{e_i}$.

$f(\cdot)$ is the feature-rich feature function of w_i.

$d(start_i - end_i - 1)$ is the distance-based reordering model based on distance to end position of previous input phrase.

v is the set of selected tokens from vocabulary V.

C_t is the contextual information for token w_i.

$\hat{l}_t$ is the feature vector derived from information within the source title set T.

$\hat{s}_t$ is the feature vector derived from information within the current sequence v^d.

$STE(\cdot)$ is the Sequence Transition Estimator (STE).

Equation 3.20 omits the token re-ordering since token ordering d is defined by the language modeling derived from source title set T. Moreover, we postulate $STE(\cdot)$ to be the feature-rich estimator for the token semantic relevance. This function should account for the token semantic information including its positional information, token occurrence information, etc. We shall decompose $STE(\cdot)$ in the next section.

Feature function f Feature function constitute of both sequence and source title set information. Intuitively, each token within the source title set should carry different degrees of saliency. This is given by its Inverse Document Frequency (IDF) within source title set T. Moreover, token's positional information should also serve as an indication as to whether it should be added next in the sequence expansion process. We also found that a binary feature indicating if a token is before an EOS tag helps to improve the overall generation. Note that all these features are language independent, and so could be employed to generate titles in a multi-lingual setting.

Our handcrafted features provide the practical benefits of allowing the machine learning models to generalize with less data. There is less of a need for large training set for the model to learn to make predictions since handcrafted features manually cover some dimensions of

Table 3.1: Decoder Sequence Features

Label	**Sequence Features**	**Description**
s_1	Language modeling (2-gram) (LM)	Bigram language modeling.
s_2	Inverse-Document Frequency (Idf)	Inverse-Document Frequency
s_3	Average token position	Mean token position within T_p.
s_4	Current Sequence Length	Number of tokens in the current sequence.
s_5	Ends with $< /s >$	Binary value to indicate if the sequence ends with </s>.
l_1	Source title set size	Number of titles within T_p.
l_2	Average Title Length	Average length of titles within T_p.

the feature space based on our domain knowledge on the listing. Let $X = w_1^t \cdots w_{|t|}^t$, we decompose the feature function $f(\cdot)$ (similar to the one in Log-linear Model) for sequence t in the following way:

$$\begin{aligned} f(X|C_t) &\approx \lambda_1\ f_1(X)\ + \cdots + \lambda_n\ f_n(X) + \lambda_1\ E(X) \\ &\approx \lambda_1\ f_1(X)\ + \lambda_2\ f_2(X) + \ \lambda_1\ E(X) \\ &= \sum_i \big(\lambda_i\ l_i(X)\ +\ \lambda_1\ s(X) + \lambda_1\ E(X)\big) \end{aligned} \tag{3.21}$$

where

l_t is a feature function derived from information within the source title set T.

s_t is a feature function derived from information within the current sequence X.

$E(X)$ is the external knowledge outside the source title set T.

However, instead of taking the weighted sum of all the feature functions, we keep it in a vectorized format as $[\hat{l}_t; \hat{s}_t]$ as input to function $STE(\cdot)$. The decoder algorithm requires an estimation for the function $STE(\cdot)$. Given a sequence of tokens t, recall equation 3.22:

$$t_p \approx \arg\max_{t \in T'} \big(STE(\hat{l}_t, \hat{s}_t)\big) \tag{3.22}$$

Function STE($\cdot$) takes an input of two vectors, $\hat{l}_i$ and $\hat{s}_i$. Respectively, a source-title-set-specific feature vector and a sequence-dependent feature vector, respectively. The function then makes a judgment about the quality of sub-sequence t. we can interpret the idea of *quality* of the sequence t as the measure of suitability for it to be used as a title.

We intend to extract the hidden information from these feature vectors via the layers of a neural network. This is done with a Multi-Layer Perceptron (MLP) with dropout. Mean Square Error (MSE) is used as the cost function, which measures of the margin between expected BLEU between gold title G and the generated hypothesis t_p and that of the predicted BLEU. The estimation of $STE(\cdot)$ is also explored with different machine learning approaches including Linear Regression (LR), Random Forest (RF), and Support Vector Machine (SVM) [18, 56, 66].

Loss Function We have explored with different loss functions including Mean square error (MSE) between y^n and $\hat{y}^n$ which is given by

$$L_{\text{MSE}} = \frac{1}{N}\sum_{n=1}^{N}(y^n - \hat{y}^n)^2$$

The idea is that the error approximated by MSE will be minimized as the weights in different layers of the MLP are updated via gradient descent algorithms as in the following general form

$$w(t+1) = w(t) + \lambda p(t)$$

and the gradient $g(\cdot) = \frac{\partial L_{\text{MSE}}}{\partial w(t)}$ at time t will then be used to update the weights.

Optimizers The Adam optimizer [47] is used as the stochastic objective function for all of the experiments involving neural network training. Its property of adaptive learning rate and relatively faster convergence allows for fast prototyping and experimentation.

3.3.3 Stack-decoding

In this section, we provide the stack-decoding algorithm and implementation details for title generation. Stack decoding is employed as the method of choice to maintain the search space. It transforms the initial search space by eliminating as much noises as possible as it gives preference to better quality sequences. This allows a likely, yet not guaranteed, conceptual attainable search space expansion from S_B (dotted line circumvented region) in the direction of the reference title space (R) to become S_E (gray shaded region). The general idea can be illustrated in Figure 3.2.

Algorithm We employ the standard beam search algorithm which is a process including the construction of the output sentence in sequence from left to right, and also the computation of transition probabilities as the hypotheses are expanded. In other words, we create the decoding process via hypothesis expansion starting with empty hypothesis. Two kinds of pruning are available − 1) histogram pruning, and 2) threshold pruning. Histogram pruning keeps a constant number of sentences while threshold pruning keeps a variable amount of sentences, depending on the distribution of sentence quality. While the search process could be exponentially huge given a full-blown breadth-first search, especially as the source title set size increases, we reduce the computational complexity by adopting the standard stack decoding with both histogram pruning and threshold pruning.

With every new token added to a sequence, the decoder will predict a new score based on the new features. This score allows the stack decoder to prune the paths and maintain only the top 10% highest-scoring hypotheses at each stack. The general idea of the entire algorithm is elucidated in algorithm 1.

Time Complexity Analysis Algorithm 1 shows the process of stack decoding with threshold pruning. While histogram pruning has a time complexity of $O(B \times d)$ where B is the beam size and d is the maximum possible length of the token sequence. However, threshold pruning differs in that the beam size B is not fixed. It has the upper bound of

Algorithm 1 Stack Decoding with Threshold Pruning

```
1: procedure STACK DECODING( threshold-size)
2:     Initialize an empty hypothesis set HypSet
3:     Initialized HYPS is a stack of 1-token hypotheses
4:     for i = 0 to |∞|  do
5:         Initialize stack S_h
6:        while HYPS ≠ empty  do
7:            pop h from HYPS
8:            expand_h ← Expand-Hypothesis(h)
9:         S_h ← Prune-Hypothesis(S_h, threshold-size)
10:        HYPS ← S_h
11:        Store hypotheses of expansion into HypSet
12:  SortedHypSet ← Sort-Hypothesis(HypSet)
13:  SortedHypSet ← End-With-EOS( SortedHypSet )
14:  SortedHypSet ← Top-20-Closest-Length-Title( SortedHypSet )
15:  Highest-scoring-hyp ← Pop(SortedHypSet)
```

the total number of tokens w present in the source title set T and a lower bound of 1 which implies that $O(1 \times d) \leq O(B \times d) \leq O(w \times d)$. Since the total number of vocabulary in any source title set is finite (it is empirically found to be proportional to the average length of title).

Pruning We also introduces pruning to be a part of the algorithm, as it determines the approach through which candidate titles are kept. For histogram pruning, we set the size of beam to be 10. Our preliminary results showed that this pruning approach yields poor performance. This is mainly due to the reason that hypotheses have a small variance of scores, and that many of them turn out to have similar scores. At times, histogram pruning would exclude the better candidates as it selects a fixed number of candidates. Instead, we explored with different thresholds 1, 0.9 as part of our parallel approach which we will further elaborate.

Desired Length Desired length D is the length of the gold title, G. It is undetermined at test time. G is also used to for heuristic selection of candidate hypotheses at line 14 of the algorithm below. Although we also explored with different desired length calculations, for the current approach we set the desired title length to be the minimum of all the title lengths within the source title set. This decision to set title length D to be the minimum stems from our empirical studies. We will substantiate this point in chapter 5.

EOS Restriction Further, we impose the EOS-based restriction on the candidate hypotheses. That is to say, we are only considering generated titles that end with the $< /s >$ tag as last token. Though this restrict generated titles to terminate in the same manner as titles within the source title set, this weeds out titles with unnatural endings and reduces dependence on the decoder.

Approach Variations There are two main parallel variations that we are investigating pertaining to this approach. One is the Greedy Stack Decoder (GSD) and the other is simply

Table 3.2: Approach Variations

Variation	Threshold
Greedy Stack Decoder (GSD)	1.0
Stack Decoder (SD)	0.9

the threshold-relaxed version of the Stack Decoder (SD). The only difference between the two models is the threshold value used for pruning.

Thus, both GSD and SD have the same algorithm, and the subsequent changes such as algorithm modifications, pruning of source title set, and re-ranking of candidate hypotheses are applied on both variations.

3.4 Discriminative Re-scoring

3.4.1 Theory

Re-scoring helps to reduce reliance on the estimator STE which directly estimates the best $t \in T'$, and also to eliminate errors resulting from the decoding process. Thus, a set of top k candidate hypotheses T', is maintained during the decoding process, before the best candidate is finally chosen. The last selection process is done in a way resembling of the task of Quality Estimation (QE), which investigates about the automatic methods for estimating the quality of machine translation output at run-time, without relying on reference translations.

Recall that previously we proposed the Sequence Quality Estimator (SQE), and let T' be the set of all the possible title generations. Thus the title t_p would be given by:

$$t_p = \arg\max_{t_h \in T', d} SQE(t_h|T, O) \tag{3.23}$$

Therefore, the resulting formulation indicates the need for the proposed model to estimate $SQE(\cdot)$. Essentially, the task would be to select a title t_p that maximizes the sequence

probability which is computed with the function $SQE(\cdot)$. The idea of sequence probability is postulated to be the BLEU score between the sequence and the reference for the original source title set T. We will now go in-depth into the proposed neural-based SQE in the next section.

3.4.2 Sequence Quality Estimator SQE(·)

Recall that the function SQE takes as input a candidate hypothesis $t_h = \{w_1, w_2, ..., w_m\}$ and outputs a quality score. Our goal is to give an estimation for the quality of the sequence with function $SQE(t_h)$. As such, the proposed idea for re-scoring is to first create the title sequence representation S_h, and then contextualizing S_h with source title set information C, before finally making quality prediction. We postulate that the final representation to be a composition of the entire state history. The overall architecture of the model is detailed in the following section.

Model Architecture

Our model encode sequence states in a token-wise manner taking as input $\{w_1, w_2, ..., w_m\}$ and convert to sequence encoding vectors $\{s_1, s_2, ..., s_m\}$. This is then fed into a Bi-directional Recurrent Neural Network (Bi-RNN) with attention. To account for the global, non-sequential information, we further concatenate the additive output from the attention layer with the global feature vector. Lastly, the resulting vector is passed into a Multi-Layer Perceptron (MLP), outputting a final value to be the estimated BLEU score. The entire model architecture is shown in Figure 3.4. Mathematically, our re-scorer model is the function $SQE(\cdot)$ that takes as inputs $\hat{l}_i$ (features derived from current sequence v^d), $\hat{s}_i$ (features derived from source title set T), $\hat{e}_i$ (token embeddings), and C (global context features):

Table 3.3: Per-layer complexity for different layer types. n is the title sequence length and d represents the number of dimensions.

Layer Type	**Complexity per Layer**	**Sequential Operations**
Recurrent	$O(n^2 \times d)$	$O(n)$
Attention	$O(n \times d)$	$O(1)$
Multi-layer Perceptrons (MLP)	$O(n \times d)$	$O(1)$

$$\begin{aligned} & SQE(t_h) \\ & \approx SQE(\hat{l}, \hat{s}, \hat{e}, C) \\ & \approx g(S(\hat{l}, \hat{s}, \hat{e}); C) \end{aligned} \tag{3.24}$$

where:

$g(\cdot)$ is a function that represents layers of multi-layer perceptrons (MLPs).

We also show the complexity for each layer in Table 3.3. Note that the number of layers of MLP is empirically determined. That is to say, we leave n in the following equations to the empirical determination:

$$g_n \left(\cdots g_2 \left(g_1 \left(S(\hat{l}, \hat{s}, \hat{e}); C \right) \right) \cdots \right) \tag{3.25}$$

Each $g_i(x)$ represents the operation $W^x x + b^x$, where W^x is the weight matrix that is a trainable tensor of $R \times |x|$ dimensions. b is the trainable bias vector of $1 \times R$ dimensions.

Next, we will further elaborate on the idea of sequence state and the details of its encoding.

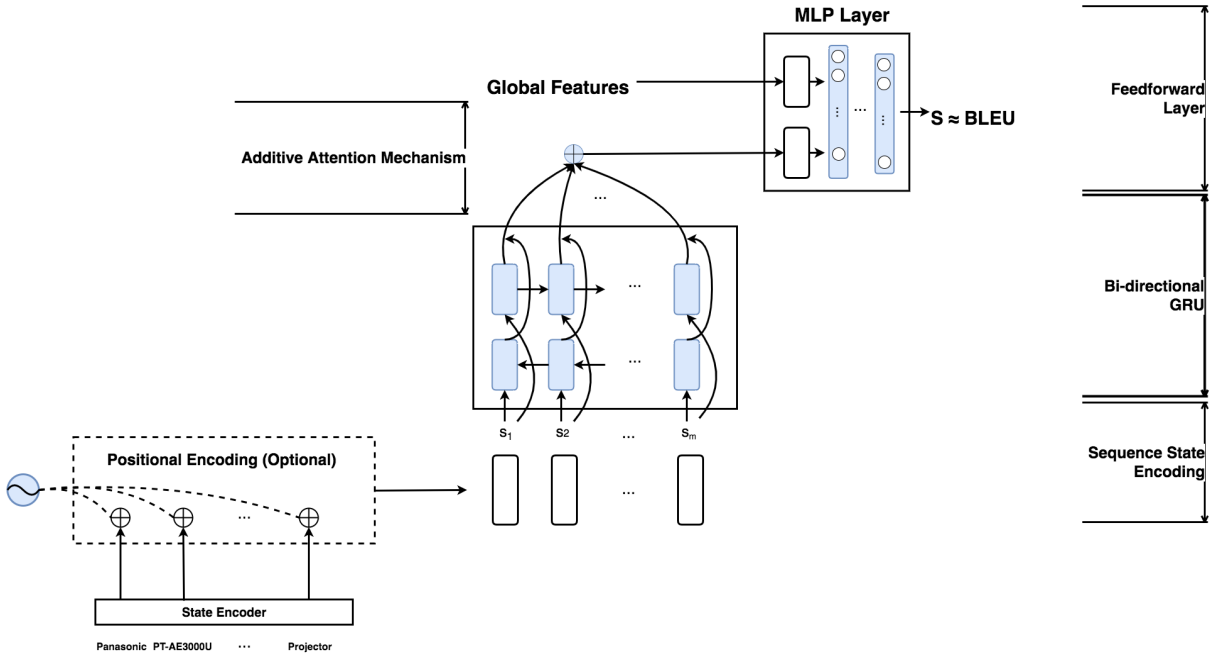


Figure 3.4: Re-scoring Model: trainable tensors in **light blue**.

Sequence State We view the sequence of tokens as a sequence of states. Each sequence state corresponds to a time step t during the process of hypothesis expansion. The sequence probability can be decomposed into

$$\begin{aligned} & p(w_1, \cdots, w_T) \\ & = p(w_1)\ p(w_2|w_1)\ p(w_3|w_1, w_2) \cdots p(w_T|w_1 \cdots w_{T-1}) \end{aligned} \tag{3.26}$$

where w_i carries the idea of a token, but in our context, also a sequence state s_i. Alternatively, we express the sequence probability as

$$\begin{aligned} & p(s_1, \cdots, s_T) \\ & = p(s_1)\ p(s_2|s_1)\ p(s_3|s_1, s_2) \cdots p(s_T|s_1 \cdots s_{T-1}) \end{aligned} \tag{3.27}$$

While the objective is to obtain title sequence representation S, the challenge is to obtain each of the conditional probabilities in the above equation. This is especially difficult given

the limitation of our data. We employ a Recurrent Neural Network (RNN) to estimate $p(s_1, \cdots, s_T)$ so that the stochastic process of Title Generation (TG) can be approximately modeled. In this way, we are not limiting the $SQE(\cdot)$ with Markov assumption, and there is now a notion of sequence history. In other words, the model now has the capacity for long-term dependency, which is the main advantage that recurrent neural networks have over models with the Markov assumption.

In theory, RNN would be able to learn the entire feature vector sequence by updating its trainable weights. A generative RNN outputs a probability distribution over the next element of the sequence, given its current state h_t, and this generative model could also captures a distribution over sequences of variable length. We will now define explicitly the constituent of each state s_t.

Recurrent Neural Network Two popular RNN architectures exist − Long short-term memory (LSTM) [42] and Gated Recurrent Units (GRU) [14]. Other than our preliminary experiments, studies have shown that GRU outperformed the LSTM on most tasks [100]. Thus in our experiments, we employ the use of GRU as our primary RNN architecture. Unlike the LSTM cells, GRU cells have no separate memory state. A GRU cell has an update gate and a reset gate. For time step k, they are given as:

$$\begin{aligned} update_k &= g(W_{update} s_k + U_{update} h_{k-1} + b_{update}) \\ reset_k &= g(W_{reset} s_k + U_{reset} h_{k-1} + b_{reset}) \end{aligned} \tag{3.28}$$

where:

$g(\cdot)$ is a non-linear activation function.

As shown in Figure 3.5, the GRU cell has its input h_{k-1} from previous states scaled by the reset gate; the update gate takes the weighted sum of the previous state and the interpolated combination of the previous state and the input s_k. The trainable weights in

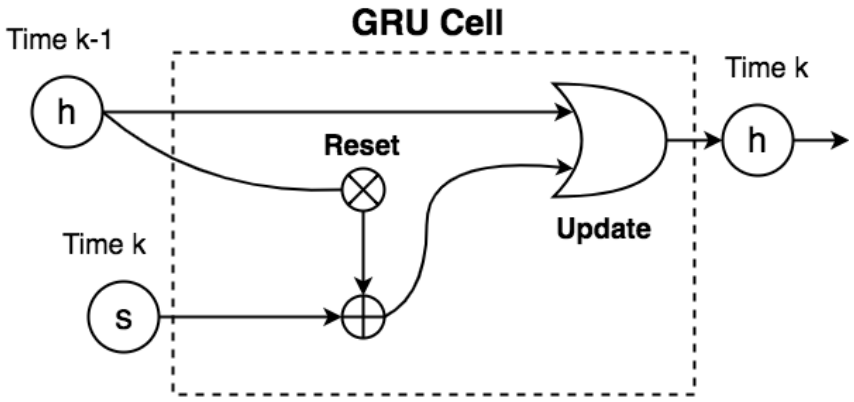


Figure 3.5: GRU Cell

both reset and update gates allow bypassing of the input or the previous states. This allows for the recurrent neural network to learn long dependency between sequence states. We will now describe the encoding function for each token within the source title set.

Token Embedding e One major practical consideration for the problem is the number of Out-Of-Vocabulary (OOV) counts that might increase dramatically at test time. However, since we are adopting an unsupervised approach, the OOV counts will thus be reduced to none. The complete statistics for the dataset will be discussed in chapter 4. All in all, the goal of embedding function e is to provide a vectorial representation for each word w_i. we employ the $fastText$ toolkit as a way to approximate this vectorial representation [7]. $FastText$ includes a sub-word model that also takes into account the internal structure of words. The model allows sharing of representations across words, which enable it to learn reliable representation for rare words. Some examples of rare words include metric-based tokens like $3.4mm$, $1.20mm$, etc, essentially product-specific tokens that only occur in source title sets for individual users. This is challenging especially when we want to utilize these vectorial representations to make predictions. However, as we will see later in the experimentation section, we are able to train effective token embedding that account for even rare words.

We propose the function $S(\cdot)$ to contain Bi-directional Recurrent Neural Network (Bi-GRU) [17, 80]. This consists of both forward and backward hidden states $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ at time t:

$$\begin{aligned} \overrightarrow{h_t} &= g(\overleftarrow{h_{t+1}}, s_t) \\ \overleftarrow{h_t} &= g(\overrightarrow{h_{t-1}}, s_t) \end{aligned} \tag{3.29}$$

This allows information from other sequence states to be shared and utilized at any time step t. Furthermore, the fact that not all tokens should contribute equally to the sentence means that there is a need to reward salient words that are essential to correctly predict a score. As such, we adopted for our re-scorer the attention mechanism.

Positional Encoding Moreover, we were also interested in seeing how a learnable positional vector could be effective in accounting for the token's individual position. As such, we also proposed to include the information of token's position into part of the sequence state embedding. To do so, we encode token position with a constant size embedding that is described by the following equation [89]:

$$E_{(pos,si)} = sin(\frac{ave\_pos}{10000^{2i/d_{pos\_emb}}}) \tag{3.30}$$

where:

i is the sequence representation.

ave_pos is the global feature function.

d_{pos_emb} is the number of dimensions within the positional embedding.

In this formulation, each dimension of the positional embedding corresponds to a sinusoid, and the sinusoidal function allows the model to extrapolate to different sequence lengths [89].

As suggested in the previous work [89], we take the sum of the token embeddings and that of positional embeddings. Moreover, we did this without removing the average token position from the feature list, in order to see if this addition could refine the existing model.

Attention Mechanism As mentioned, we need a way to weigh differences in saliency. This comes in the form of a rewarding mechanism is realized in the form of the attention mechanism function. Of the various forms of attention mechanisms, we employ the global attention [58] where association score is calculated sentence-wide and output at each time step of the RNN is weighed accordingly. The main motivation behind an attention layer is to compute an association between the contextual vector and sequence state encodings within the sequence. Between contextual vector A and another sequence state encoding s_k at time k, the raw association is computed as such:

$$a_j = s_j A^T \tag{3.31}$$

We then normalize the association using a softmax function:

$$\alpha_j = \frac{\exp(a_j)}{\sum_k \exp(a_k)} \tag{3.32}$$

Since we do it for every time step, we will have a total of m association computations. The last remaining step to obtaining the context vector $S(t_h)$ is the following:

$$S(t_h) = \sum_j \alpha_j s_j \tag{3.33}$$

Combining the intuition for the context vector, we now have $SQE(\cdot)$ as

$$\begin{aligned} SQE(t_h) &= h(\ [S(t_h);\ C]\ ) \\ &\approx h(\ [S(t_h);\ g(t)]\ ) \end{aligned} \tag{3.34}$$

Table 3.4: Re-scoring Sequence State Features

Label	Sequence Features	Description
s_1	Language modeling (2-gram) (LM)	Bi-gram language modeling.
s_2	Inverse-Document Frequency (Idf)	Inverse-Document Frequency
s_3	Average token position	Mean token position within T_p.
s_4	Current Sequence Length	Number of tokens in the current sequence.
s_5	Ends with $< /s >$	Binary value to indicate if the sequence ends with </s>.
e_1	Token embedding	50-dimensional token embedding trained on the entire source title set corpus.

Table 3.5: Re-scoring Global Features

Label	**Global Features**	**Description**
l_1	Source title set size	Number of titles within T_p.
l_2	Average Title Length	Average length of titles within T_p.

where:

$S(t_h)$ is the sequence representation.

$g(t)$ is the global feature function.

Features we will now expand on the idea of the two features

1. Sequence state features, $[\hat{s}_1; \hat{e}_1]$ and

2. Global features, $[\hat{l}_1]$

Sequence state features are what constitutes the notion of states, while global feature functions accounts for the high-level features in order to include more information outside of the sequence. The two types of features are indicated in Table 3.4 and 3.5, respectively.

Both our decoder and our re-scorer adopt more or less the same set of features. However, the difference lies in the way they are used, depending on the nature of each of the features. $STE(\cdot)$ takes both token-specific and source-title-set-dependent features jointly; the re-scorer imposes strict constraints on their respective usages. Specifically, the re-scorer feeds in token sequence features sequentially to the RNN and concatenates the output vector with the global feature. This goes with our intuition that source title set features are constant throughout the entire sequence and thus having them in the sequence state would not make sense.

Loss Function We have also attempted with several variations of loss functions including L1-norm, Mean Absolute Error (MAE). We found no major differences in the performance but that L2-norm yields a relatively stable performance. As such, we adopt the same loss function as that for the function $STE(\cdot)$, consisting of the L2-norm, or the Mean Square Error (MSE). The loss between the estimated quality score $SQE(t_h)$ and target BLEU $BLEU_t$ is given below:

$$Loss(SQE(t_h), BLEU_t) = \sum_i \frac{1}{2}(BLEU_t - SQE(t_h))^2 \tag{3.35}$$

Additional Refinements On the other hand, we added additional dropout layers to the $SQE(\cdot)$ model − one to the output of attention layer and the other to the output of concatenation between sequence state feature vector and global feature vector. We use a ratio of 0.2 for both dropout layers.

For each mini-batch, we perform the batch normalization as a form of regularization. It makes the training process more robust against poor initialization of weights and more tolerant to different learning rates.

3.5 Conclusion

To conclude, we propose a Generative-Discriminative framework consisting of a stack decoder and an attention-based neural re-scorer that serves to complete the entire pipeline from generation to the final best-title selection. The process is loosely a form of boosting in machine learning where two independent classifier is learned sequentially on separate data distribution. As such, the initial bi-gram forest is pruned by the stack decoder to allow for selection by the re-scorer. In the next few chapters, we will finally embark on the empirical explorations by first introducing the datasets and then empirically establish the need for the stack decoder, and the effectiveness of our proposed framework.

Chapter 4

DATA SETS

In this chapter, we describe the dataset used. We obtained an annotated e-commerce title dataset that consists of about *80,000* source title sets. Each source title set consists of: (1) a set T of source titles (user-created sequences of words that describe a product), (2) a title, i.e. a human-curated title that describe the product, which is taken to be the reference or gold title that we attempt to generate.

Source title set size is the number of source titles for each product, basically it is the length of T, or $|T|$. In this chapter we will first explain in detail how the dataset was prepared, then we will talk about how it was divided into the training and testing sets, and finally concluding with their respective statistics.

4.1 Data File Formats

All the data is first converted to a JSON format. This format assigns a unique ID to each of the products and also to all of the source title sets. This allows ease of querying the source title sets based on its product ID, which is the unique identifier for each product, and so no two IDs could belong to a single product and vice versa. One example of a product in JSON is displayed in Figure 4.1.

As shown in Table 4.1, the distribution of products with source title sets of varying sizes is not uniform. There is a greater amount of source title sets with low set sizes, or simply, most source title sets have a small number of associated user-created titles. This idea is clearly illustrated in Figure 4.2, where source title set sizes and count for each size are inversely proportional.

```
{
   "Product": {
      "ID": "0",
      "gold": "QNT TESTEK Testosterone Growth Optimizer 120 Capsules",
      "items": {
         "0": "QNT TESTEK Testesterone Growth Optimizer, Capsules",
         "1": "Qnt International Testek",
         "2": "Qnt - Testek - 120 caps",
         "3": "Testek Testosterone",
         "4": "QNT TESTEK Testosterone Growth Optimizer, Capsules",
         "5": "TESTEK 120 Capsules",
         "6": "QNT TESTEK Testosterone Growth Optimizer, Capsules",
         "7": "TESTEK 120 Capsules",
         "8": "QNT TESTEK Testosterone Growth Optimizer, Capsules, 120 ea",
         "9": "Qnt International Testek - 120 Rapid-Release Capsules"
      }
   }
}
```


Figure 4.1: Sample product JSON collected

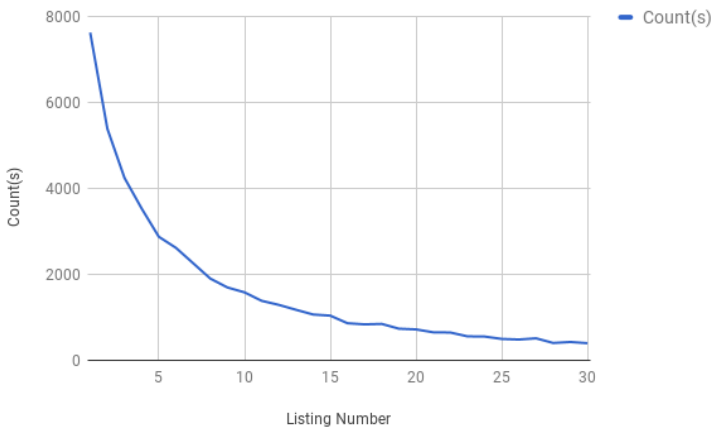


Figure 4.2: Plot of source title set size w.r.t Counts

Table 4.1: Source title set size w.r.t counts

Set Size	Count	Set Size	Count
1	7615	16	863
2	5382	17	836
3	4239	18	847
4	3528	19	736
5	2872	20	719
6	2614	21	652
7	2258	22	647
8	1901	23	558
9	1694	24	553
10	1580	25	498
11	1383	26	484
12	1289	27	512
13	1175	28	404
14	1065	29	427
15	1039	30	399

4.2 Data Sets

Our dataset contains both English and German data. Other than the final test for multi-linguality, we will perform most of the experiments on the English dataset. We will now detail the usage of the English data.

We intend to test the models' ability to generalize with different data sizes. As such, we created two alternative testing sets. The common 80:20 split between training and testing set is ensured and source title sets with different set sizes are evenly distributed. The other would be the larger testing set which includes additional range of source title set size from 2 to 30. The idea is that the model will have to be able to generalize to large amount of unseen data (source title set sizes in the range of 2 to 30) after learning with very little data (source title set size in the range of 2 to 10).

We decided that source title sets with only one user-created title, or equivalently, products with source title set size of one, will be neglected. This is done for two reasons. Firstly, title sets of size one could simply be chosen as the target title without resulting in biased performance. The problem that originally motivates this work would be invalid in this context. If there is no other users for the product, there will not be any "biases" for this title as a singleton.

Since source title set sizes are roughly inversely proportional to the their counts, we opted to select from sets with the smallest sizes, since they are more easily available in the practical sense − e-Commerce companies could get their hands on products with these source title set sizes more conveniently. As was mentioned, larger title sets will have lower occurrences. In fact, set counts decrease to double digits at set size of 68. we constrained the training set to source title sizes from 2 to 30 so that the results from the thesis can also be easily applied to the real world scenarios. To be precise, products with smaller source title set sizes are more accessible than those with greater set sizes. We found that most products naturally have low source title set sizes. This means that in most cases, products have few (2-30) corresponding source titles. This phenomenon becomes more prominent as source title set size increases to

Table 4.2: Datasets

	Train-210	*Test-210*	*Test-230*	*Test-random*
Title Set Size Range	2-10	2-10	2-30	Random
Number of Source Title Sets	3,600	900	2,900	400

100, where less than 50 products with source title set size greater than 100 exist.

In order to test the generalizability of our models to varying source title set sizes, while taking into account the source title set accessibility, we selected our training set from source title set with sizes ranging from 2 to 10. As such, 400 source title sets between set sizes from 2 to 10 were selected to form the training set, $Train$. On the other hand, the two test sets were selected to test the robustness of the model. They are namely, *Test-210*, *Test-230*, and *Test-random*. *Test-210* comes from set sizes 2 to 10, with 100 products for each set size, while *Test-230* would be the same collection of 100 title sets between sizes of 2 to 10, but with additional collection of 100 source title sets of sizes between 11 to 30. Lastly, *Test-random* comes from a randomly selected collection of 400 source title sets with various set sizes. In particular, *Test-random* is subjected to human evaluation so as to determine the existence of model errors. Note also that *Train* and *Test-210* form the 80:20 split. For the above reasons, we divide our datasets as shown in Table 4.2.

4.3 Data Sets Statistics

At this point, we have covered the reasoning for the data set construction. We will examine the datasets even further so as to figure out the pre-existing settings within the source title sets. Before we do that, we will first define some terms for clarity. As shown in Figure 4.1, each product will have 3 essential attributes. They are the product ID, *gold* as in the reference title, and the *sourcetitleset* T, as was previously mentioned. Moreover, each source title set t_i within T will have an ID, giving us the convenience for error analysis.

Table 4.3: Statistics of *Test-210*

Source Title Set Size	Mean Gold Title Length	Mean *Oracle* BLEU	OOV Count
2	5.74	0.593	41
3	6.36	0.674	37
4	6.44	0.692	30
5	6.35	0.74	29
6	6.89	0.71	33
7	6.96	0.759	31
8	6.87	0.802	24
9	6.73	0.776	25
10	6.8	0.763	17
Ave.	6.57	0.723	29.7

With that said, we will now examine the datasets. For dataset $Train$, we examine the mean gold title length, mean Oracle BLEU, and the OOV count among all the source title sets T. Oracle BLEU is the highest quality title among all source title sets in terms of BLEU scores. OOV count is the number of tokens that are present in the gold title, but are absent in source titles T.

As we can see from Table 4.3, the mean gold title length stays relatively constant despite the increase in the size of source title set. On the other hand, the *Oracle* BLEU has a positive correlation with the size of source title set, while OOV count displays a decreasing trend throughout the range of size of source title set. As size of source title set increases, the likelihood of finding tokens belonging to the reference title among the source titles is higher. This also implies that it is easier for *Oracle* to find a good title.

Moreover, mean oracle BLEU displays a positive correlation in the sense that higher sizes of source title set will yield better *Oracle* performance. As we will see later, this is

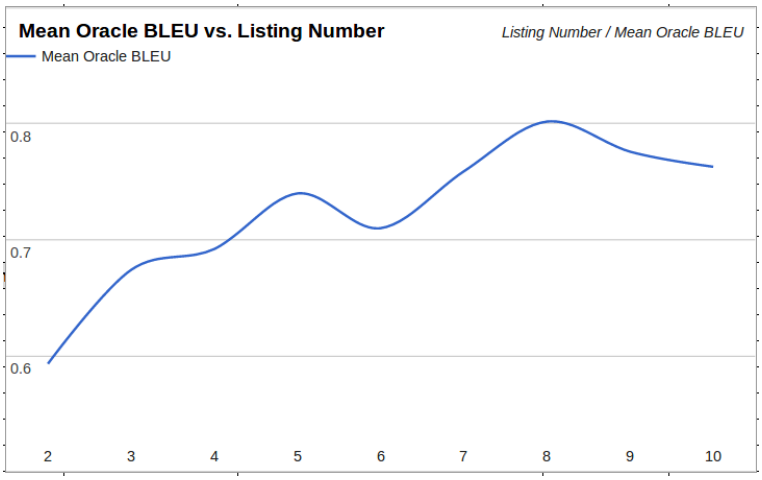


Figure 4.3: Change of *Oracle* BLEU against Source title set size

consistent with our model performances. This behavior is captured in Figure 4.1 which shows an upward trend.

In the next chapter we will discuss in-depth our experimentations on all the datasets including *Train-210*, *Test-210*, *Test-230*, and *Test-random*. The main idea is that we will train our models on *Train* and measure the model performances against all the testing sets.

Chapter 5
EXPERIMENTAL RESULTS

In this chapter, we will detail the experiments that we carry out to investigate the different components of the model with the following four objectives

Multi-linguality Models exhibit clear trend of ability to perform in a bi-lingual setting (German and English datasets) in our experiments.

Generalizability Models trained on a small dataset is able to perform similarly when tested on a large dataset.

Robustness When the experimental settings such as the size of source title set is varied, models perform competitively.

Tractability Algorithms for the system are constrained within polynomial time.

Overview of Presentation The order of our experimentation will be extended in the following manner. First, we will explore the framework components by examining both the decoder and the re-scorer separately. We will first analyze $STE(\cdot)$'s learning capabilities and the decoder's configurations. To do so, we will compute the learning curve for $STE(\cdot)$ as the BLEU estimator used within the decoder by incrementally increasing the data sizes. Our intention is to justify the use of the neural network over the baseline log-linear model as a way of estimating $STE(\cdot)$. Next, we explore the effect of varying decoder algorithm's configurations on the decoder's performance. We propose a *Base algorithm* to explore the effect of desired title length D (minimum, maximum or mean of all source title lengths) and token average (mean, mode, and median) on generation. These results would influence our

decisions for the choice of features, and also inspire the algorithmic adjustments that we made with regard to the stack decoding algorithm. We also study the importance of each of the features by computing the feature importance scores. Lastly for the decoder, we want to verify the claim that the framework is indeed generation and BLEU-boosting. Comparison is made between the variance and mean of BLEU among titles before and after decoding.

Having established that, we will then explore the different components within the proposed re-scorer. This is done by first showing our preliminary discussion on the bi-lingual token embedding. We show that rare words and sub-word information are both accounted for in token's vectorial representations.

The last part of our experimentation will involve explorations on the entire framework consisting of both decoder and re-scorer. This is where we intend to establish the final claims that the models are robust, generalizable, tractable, and able to function in a bi-lingual setting. At this point, we will define all the models that will be used in this chapter (shown in Figure 5.1).

Evaluation Metric Note that, although our preliminary explorations involve multiple metrics such BLEU (bilingual evaluation understudy), TER (Translation Error Rate), and ROUGE (Recall-Oriented Understudy for Gisting Evaluation), the rest of the experiments will employ the evaluation metric BLEU since it was shown to correlate well with both TER and ROUGE [71]. Unless stated otherwise, all subsequent experiments are trained with *Train-210* and tested on different test sets including *Test-210*, *Test-230*, or *Test-random*.

Hardware and Training Details We trained and tested all our models with 1 Intel Xeon CPU. For training of all decoder models, we employ a small batch size of 32 and ran it for 100 epochs. Re-scorer models are trained with varying sizes of epochs from 32 to 10,000 in less than 50 epochs (since we are using CPU there is less of a need to make it a power of 2). Re-scorer models takes a step of approximately 150 seconds and run for 2.5 hours before convergence (via Keras with Theano backend). On the other hand, it takes less than 20

Table 5.1: Approach Summary: below is the summary of all approaches compared in this chapter. E.g. *SD+BiRNN* would simply mean a stack decoder with re-scorer that uses the Bi-RNN as shown below.

Approach		Description
Oracle	**-**	**Selection of highest BLEU-scoring title (with respect to the reference title) from among the source title set.**
Base algorithm	**-**	**Proposed algorithm that utilizes only the token frequency and average token position to generate titles.**
GSD	**LLM**	**Stack decoder (beam search threshold set at 1.0) that uses the log-linear model to estimate $STE(\cdot)$.**
MostFreq	**-**	**Algorithm that simply selects the most frequently-occurring title from among the source title set.**
MostFreq	**OpenNMT**	**Built on top of MostFreq, which selects the most frequently-occurring title from among the source title set, and then uses a trained OpenNMT model to generate a title.**
SD	**-**	**Stack decoder (beam search threshold set at 0.9)**
	BiRNN	**Stack decoder with re-scorer that uses the Bi-RNN.**
	ATT	**Stack decoder with re-scorer that uses the Bi-RNN that contains attention mechanism.**
GSD	**-**	**Greedy stack decoder (beam search threshold set at 1.0)**
	BiRNN	**Greedy stack decoder with re-scorer that uses the Bi-RNN.**
	ATT	**Greedy stack decoder with re-scorer that uses the Bi-RNN that contains attention mechanism.**

minutes with our PyTorch implementation with various batch sizes, while displaying similar results. We employ early-stopping with a patience of 10 epochs while setting the maximum epoch at 100.

Baselines In the course of our research, we worked with a few baselines including the proposed **Base algorithm**, and the **GSD + LLM**. **Base algorithm** will be introduced in greater details later on. However, it is essentially an algorithm that builds tokens based on 1) token frequency and the 2) average token position. We framework is built on top of the **GSD + LLM**, where titles are expanded token-wise and each token sequence is scored with the log-linear model with the same feature set that we used for our models. We used **threshold pruning** that selects the top $k\%$ token sequences based on their scores as given by the log-linear model.

Benchmarks As discussed in the chapter of related works, we adopted three primary benchmarks. These include **Oracle**, **MostFreq**, and the **MostFreq + OpenNMT**. Due to the lack of existing benchmark systems for comparison, we compared our system with the **Oracle**, which is the theoretical best system. It also serves as the upper bound to any selection methods employed by the companies without re-decoding. Next, we observed that reference title tends to appear the most frequently of all the titles as the size of source title set increases. Therefore, we propose to select the most frequently occurring title in **MostFreq**. Experimentally, **MostFreq** proves to be a rather competitive approach. It exploits on the human annotation process since titles were created by annotators who had access to the source titles. Next, we intend to compare our system against the state-of-the-art Neural Machine Translation (NMT) system, *OpenNMT*, which is an encoder-decoder model with attention. Ideally, **MostFreq** serves as the performance lower bound to **MostFreq + OpenNMT** so as to give our system a more competitive comparison.

5.1 Framework Baseline

The Generative-Discriminative framework consists of a decoder and a re-scorer. In the next section we will touch upon the decoder performance without re-scoring, and examine the effect on its performance when configurations are altered.

Our framework baseline is the conventional log-linear model based decoder. This is the case where $STE(\cdot)$ is estimated via linear regression in order to learn the weights λ_i for different feature functions as given:

$$p(x) = \exp(\ \sum_{i}^{n} \big(\lambda_i\ f_i(x)\big)\ ) \tag{5.1}$$

We will employ a linear regressor in order to estimate the weights λ_i for the features.

5.2 Decoder

Decoder takes as input of source title set T and output a candidate list of generated titles. As was discussed previously, a stack decoding model is used in the generation to generate candidate titles. Moreover, the decoder also assigns each generated titles a quality score with function $STE(\cdot)$.

Preliminary Experiments Before our decoding algorithm was fully developed, we have performed numerous preliminary experiments to determine the heuristic decisions in both our algorithm and our feature set. For our algorithm, this includes the **beam search threshold value** and the **title length** for the final heuristic pruning. Feature set contains the **average token position**, so we also tested with different versions of average computation including 1) mean, 2) mode, and 3) median. Moreover, effect of size of source title set is also investigated so as to obtain a good understanding of the relationship between number of titles present and performance. Lastly, we begin the section with the comparison between linear and non-linear approximation of the State Transition Estimator (STE). This paved the way for our employment of non-linear function for the STE.

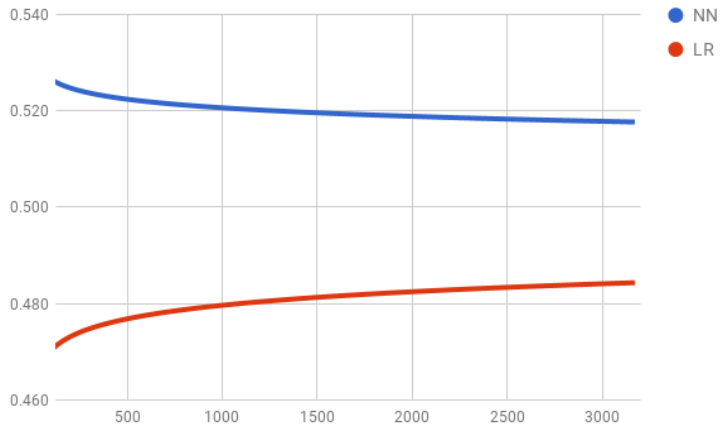


Figure 5.1: Plot of BLEU against data size: learning curves for training with linear regression and neural network

5.2.1 Learning Curve

We will now proceed to compute the learning curve for State Transition Estimator (STE) trained with both linear regression and neural network, which are shown in Figure 5.1.

The graphs are computed on five different shuffled datasets with increasing dataset sizes. We use *Test-230* as the test set while the model $STE(\cdot)$ is trained on the five different randomly sampled datasets with the same sizes. We computed the average BLEU for all five constant-size datasets to be used for the plot. As we can see from the two figures, BLEU that is represented by they Y-axis peaks at around 800 source title sets for linear regression and at 1,600 source title sets for neural network. Moreover, the highest BLEU is higher for neural network at 0.532, comparing to that of linear regression which is at 0.504. This shows that a neural-tuned STE is outperforming a linear regressor regardless of the dataset size.

5.2.2 Source title set size and Threshold

In this section, we intend to experiment with two parameters of the decoder. They are 1) beam search threshold values and 2) the desired length determination. Specifically, beam

Table 5.2: A Table of BLEU Against Data Size

Data Size	125	150	175	200	400	800	1,600	3,200
NN	0.531	0.528	0.5272	0.5178	0.512	0.5224	0.531	0.513
LR	0.464	0.470	0.474	0.473	0.473	0.504	0.474	0.477

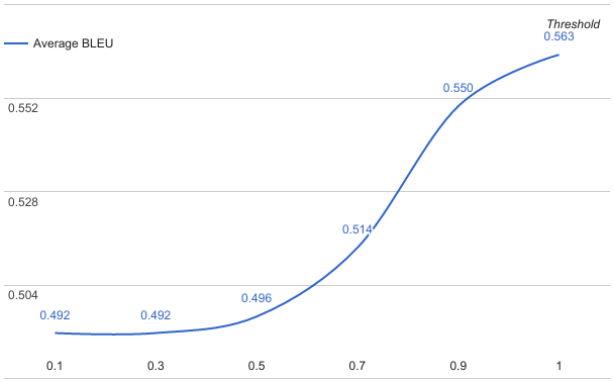


Figure 5.2: Plot of BLEU against beam search threshold: model performance increases as threshold rises

search thresholds are set to be values $\{0.1, 0.3, 0.5, 0.7, 0.9, 1\}$, and desired length could be either 1) minimum, 2) maximum, or 3) mean of source title lengths. For this experimentation, the test set will be *Test-210*, which contains sizes of source title set in the range between 2 to 10.

Table 5.3 clearly shows that low beam search threshold values produce low-scoring performances while setting threshold value to 1.0 yields close to one BLEU point improvement. We postulate the reason to be low threshold values maintain a greater amount of titles, which could potentially introduce poor quality titles. This exacerbates scenarios where high BLEU-scoring but less desirable titles are prevalent, making the final selection process difficult. Owing to this reason, we see that threshold values of 0.9 and 1.0 yield the best performances. This trend is depicted in Figure 5.2 where there is a consistent upward in-

Table 5.3: Table of threshold values against size of source title set

Threshold \ Source title set size	2	3	4	5	6	7	8	9	10	Ave.
0.1	0.539	0.568	0.555	0.52	0.483	0.453	0.431	0.435	0.441	0.492
0.3	0.539	0.568	0.555	0.52	0.483	0.453	0.431	0.435	0.441	0.492
0.5	0.539	0.568	0.555	0.518	0.486	0.463	0.434	0.452	0.448	0.496
0.7	0.539	0.572	0.558	0.541	0.517	0.465	0.488	0.454	0.488	0.514
0.9	0.538	0.589	0.588	0.55	0.546	0.518	0.528	0.537	0.555	0.550
1	0.522	0.595	0.565	0.577	0.539	0.513	0.584	0.57	0.603	**0.563**

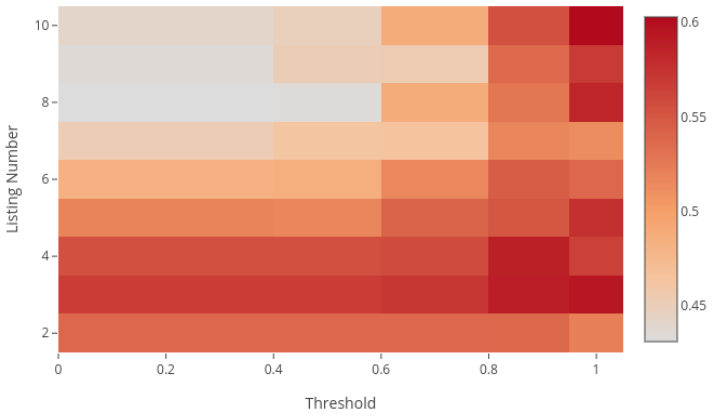


Figure 5.3: Heatmap of BLEU for sizes of source title set against threshold

crease as the threshold value goes up. We also plotted the heat map showing the BLEU score distribution across both threshold and size of source title set in Figure 5.3. Since value of BLEU is indicated proportional to the degree of redness of the region, we observe a similar relationship between threshold and model performance (BLEU).

Other than the effect of threshold on the performance, we also noticed the model is obtaining higher BLEU scores when the size of source title set is small across all threshold values. This makes sense because smaller sizes of source title set will have less possibilities of possible good expansions and thus the title distribution will have a lower mean BLEU. Again, this trend is captured in Figure 5.3 where higher sizes of source title set correspond to tiles with higher degree of redness.

5.2.3 Token Position

As shown in Figure 3.1, our decoder has the average token position as one of the features. However, there are three different ways of computing the average, namely, 1) mean, 2) mode, and 3) median. We will now investigate the effect of each of these definitions in order to select the best for our framework.

We propose a ***Base algorithm*** in which we use only token position and token frequency to generate the title. The construction of the Base algorithm is inspired by our observation that there is an intersection between all titles at the token level − and so using token-specific information for generation allows us to gain insights into how these information fit together. We outline the basic flow of the Base algorithm below:

Algorithm 2 Base Algorithm

1) *Sort tokens according to frequency*

2) *Select top-k tokens*

3) *Sort selected tokens by average positions*

As such, the purpose of the *Base algorithm* is to perform preliminary explorations based

Table 5.4: Performance of Base algorithm against token position determination methods in terms of BLEU

Source title set size	**Mean**	**Mode**	**Median**
2	0.488	0.481	0.479
3	0.483	0.491	0.478
4	0.496	0.504	0.492
5	0.481	0.465	0.464
6	0.487	0.499	0.482
7	0.469	0.475	0.467
8	0.467	0.47	0.469
9	0.485	0.486	0.48
10	0.634	0.635	0.634
Mean	0.499	**0.501**	0.494

on token frequency and average token position within the source title set. The average of token positions could be seen from the perspectives of 1) mean, 2) mode, and 3) median. In Table 5.4, we compute the BLEU scores for each token position determination methods on *Test-210*. Average as mode is shown to have the highest mean BLEU in source title set sizes in the range of 2 to 10 with a mean BLEU of 0.501.

We display the general trend in Figure 5.4. Even though the performances do not differ by a significant amount, average by mode performs competitively at source title set sizes 4 and 7 and appears to be more robust against the size of source title set. Therefore, we chose mode to be our method of computing the average token position.

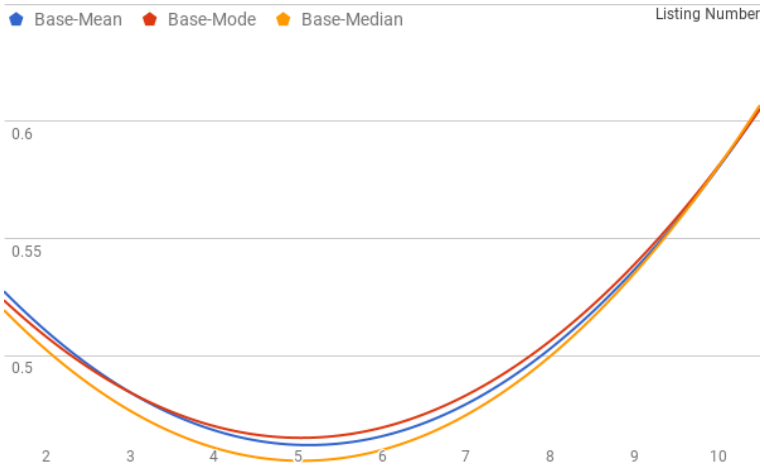


Figure 5.4: Plot of BLEU against token position average methods

5.2.4 Title Length Variations

Next, we set the threshold at 1.0 and vary the desired length. Similarly, the test set is *Test-210*. The results are shown in Table 5.5. We see that having the desired length set to the minimum length of the source title lengths would give us a consistently better performance against source title set of sizes 2-10. The mean BLEU over the range of sizes of source title set yields approximately one point improvement over when the length is set to be the maximum and mean. This motivates our choice for title length to be set at the minimum length of all source title lengths.

Table 5.5: Table of size of source title set against desired length variations

Source title set size	**Min.**	**Max.**	**Mean**
2	0.522	0.516	0.516
3	0.595	0.593	0.593
4	0.565	0.557	0.557
5	0.577	0.573	0.575
6	0.539	0.539	0.539
7	0.539	0.508	0.508
8	0.584	0.58	0.582
9	0.57	0.559	0.561
10	0.603	0.6	0.6
Mean	**0.566**	0.558	0.559

5.2.5 Feature Importance

In order to determine the importance of each features, we employ a meta estimator that fits 250 randomized decision trees on various sub-samples of the dataset obtained from *Train-210*. This estimator helps us to obtain a score for each features so as to get the individual feature importance score. The results are shown in Table 5.6 and the plot is displayed in Figure 5.5.

We found that average token position and the Inverse-Document Frequency (idf) are the two most important features. This shows that the frequency and the location of where a token occurs are significant to the decoding process. Following the two features, 2-gram language modeling (LM), current sequence length, and average title length contribute similarly. Least important features are 1) ends with $< /s >$ and 2) size of source title set. Significantly, this shows that size of source title set does not have a major contribution, which implies the idea that the decoding process is independent of source title set sizes.

Table 5.6: Feature importance

Sequence Features	**Importance Score**
(0) Language modeling (2-gram) (LM)	0.149
(1) Inverse-Document Frequency (Idf)	**0.206**
(2) Average token position	**0.236**
(3) Current Sequence Length	0.136
(4) Ends with $< /s >$	0.0499
(5) Source title set size	0.0937
(6) Average Title Length	0.129

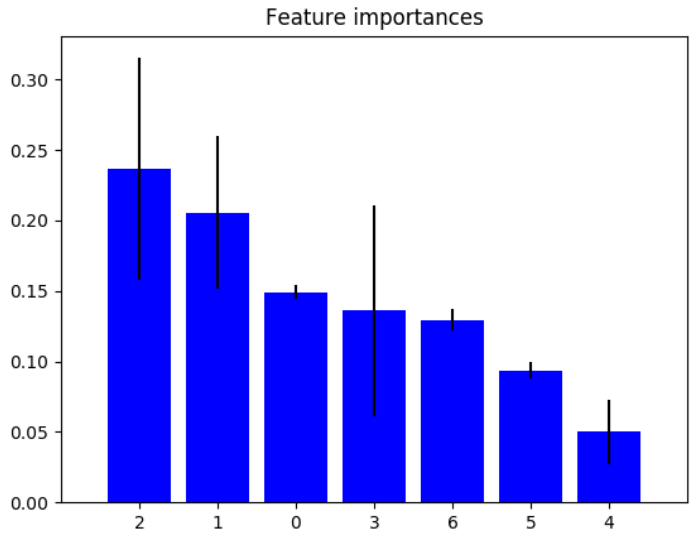


Figure 5.5: Plot of feature ranking

Figure 5.6: Plots of BLEU for varying quality of titles as defined by selecting all titles (100%), top 30% and lowest 30% BLEU-scoring titles on *Test-230*. The two plots below show that 1) pre-selection of input titles creates a huge marginal difference, and 2) *GSD+ATT* outperforms *SD+ATT* in all three scenarios of pre-selections.

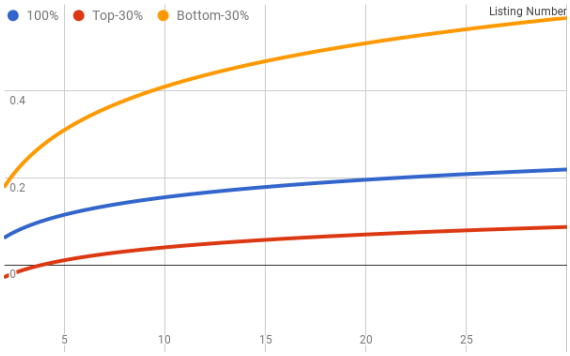


Figure 5.7: Performance for *GSD+ATT*

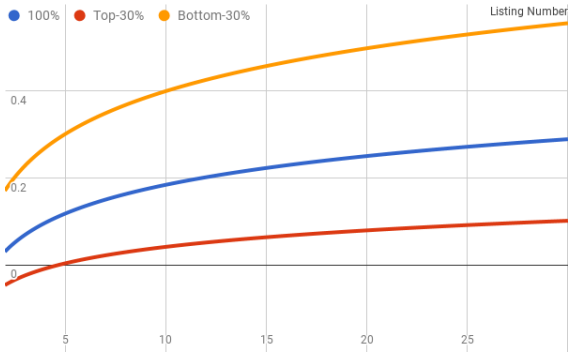


Figure 5.8: Performance for *SD+ATT*

5.2.6 Framework Reduction

In this section we will discuss about the BLEU distribution's variance reduction as we reduce the search space from B to E. As stated, we expect there to be a corresponding improvement between model performance and quality of titles.

The initial step is to determine the necessity of the generation step by selecting titles based on their qualities. By doing so, we intend to artificially create scenarios similar to

Table 5.7: Model performances (BLEU) against pre-selection

Model	**100%**	**Top-30%**	**Bottom-30%**
SD + ATT	0.622	**0.776**	0.394
GSD+ATT	0.661	**0.781**	0.383

the desired framework property − 1) limiting variance of BLEU, and 2) boosting the mean BLEU within the distribution.

We define quality of a title as its BLEU score with respect to the reference title. We will explore the outcome on both the best quality (or highest BLEU-scoring) titles and those with the worst quality (or lowest BLEU-scoring). More specifically, best quality and worst quality titles are defined as the top 30% and bottom 30% BLEU-scoring within the set, respectively. All subsequent performances are the BLEU scores between the generated title t_p relative to the reference title. The model we use here are two of our best models consisting of *SD+ATT* and *GSD+ATT*.

Table 5.7 shows that choosing good quality titles as input to the models could significantly improve the performances. The same goes for both models *SD+ATT* and *GSD+ATT*, as they both encounter a significant increase in BLEU. In fact, pre-selection of titles yields BLEUs of 0.776 and 0.781, which are significant upgrade from the full set, where all titles are considered. This result motivates our framework which aims at producing better quality titles in order to boost the framework performance.

In the following experiments, we will examine the transformation as performed by the framework on both the variance and the average BLEU for different sizes of source title set. We intend to find out if the following are true:

- Increase in search space quality (BLEU): boosting of average BLEU on search space (from B to E) across sizes of source title set.

- generation: relatively constant variance on BLEU distribution (within the same search space) across sizes of source title set.

As we can see from Figure 5.9, the framework successfully increased the average BLEU for all the possible titles. In fact, the BLEU margin increases as the size of source title set increases, showing that the framework becomes even more successful as the size of source title set rises. On the other hand, variance of the distribution of BLEU remains largely

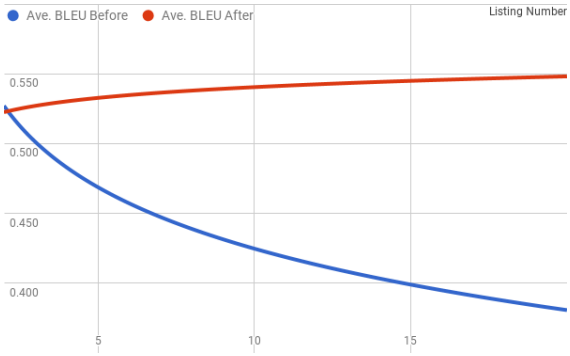


Figure 5.9: BLEU transformation

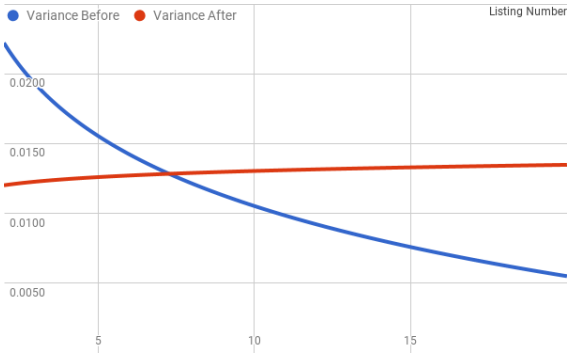


Figure 5.10: Variance transformation

similar before and after the transformation by the framework, as is shown in Figure 5.10. As such, we can conclude that the framework is indeed generation and BLEU-boosting, which tallies with our postulation about the framework.

5.3 Re-scorer

we will now talk about the experiments and results for adding re-scoring to the decoder from the previous section. From previous discussions we learned that a high threshold $[0.9, 1]$ (or keeping a smaller percentage of candidate hypotheses) during stack decoding yields better performances. A key component of the re-scoring model is the bilingual token embedding which we used as input to our neural-based model.

5.3.1 Bilingual Token Embeddings

In this sub-section, we will discuss the token embedding trained with the FastText toolkit. The aim is to show how the FastText sub-word model can be relevant to this task by providing a more reliable representation of rare words.

Token embedding is trained on the entire corpus which consists of roughly 80,000 source title sets in an unsupervised fashion using the fastText [7]. The skip-gram option is used with minimal number of word occurrence set at 2. This amounts to $13,032$ tokens and the

Table 5.8: Abbreviations descriptions

Figure	Abbreviation	Description
5.11	gb	Gigabyes. E.g. 120 gb
5.12	mm	Millimeters. E.g. 19mm
5.13	port	Connection hole found on the electronic devices. E.g. 16-port

log loss is about 1.56. All titles are preprocessed in the same manner as that in training and test time. This ensures that the OOV is 0. Moreover, tokens which are numbers (e.g. 960) are converted to their Part-Of-Speech (POS) tags as either *FLOAT_CD* or *INT_CD*. This yields a more unifying representation since many numeric tokens have single occurrences.

We will now go through the terminologies used in the following figures. In the context of titles, frequently used terms are often abbreviated. A few examples are given in Table 5.8. Figures 5.11 to 5.13 are 3-dimensional graphs plotted with the method of Principal Component Analysis (PCA) for non-linear dimensionality reduction. The plots show that the trained embedding entails meaningful representations for rare words. The reason for this is mainly due to the sub-word considerations that fastText has adopted.

In Figure 5.11, variations of *gb* are close to each other in the 3-dimensional euclidean space. In fact, tokens like *960gb* have a low occurrence within the entire corpus, but showed high similarity with other *gb*-based tokens in the figure. In Table 5.9, we show the count of some of the tokens found within Figure 5.11.

Thus, we conclude that the sub-word model has learned meaningful token representations from a limited set of data, even when some given tokens are of low occurrences. This is especially important for our task where many tokens are mis-spelled by users.

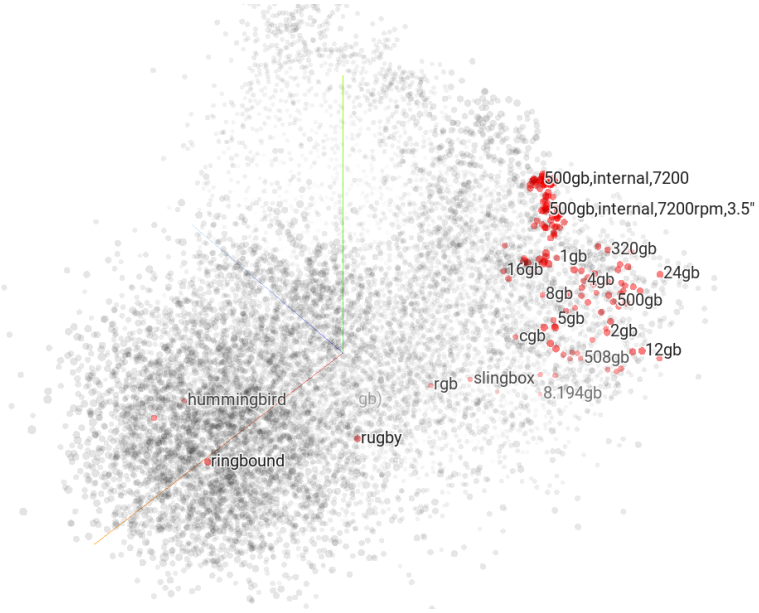


Figure 5.11: Three-dimensional euclidean projection for tokens related to *gb*

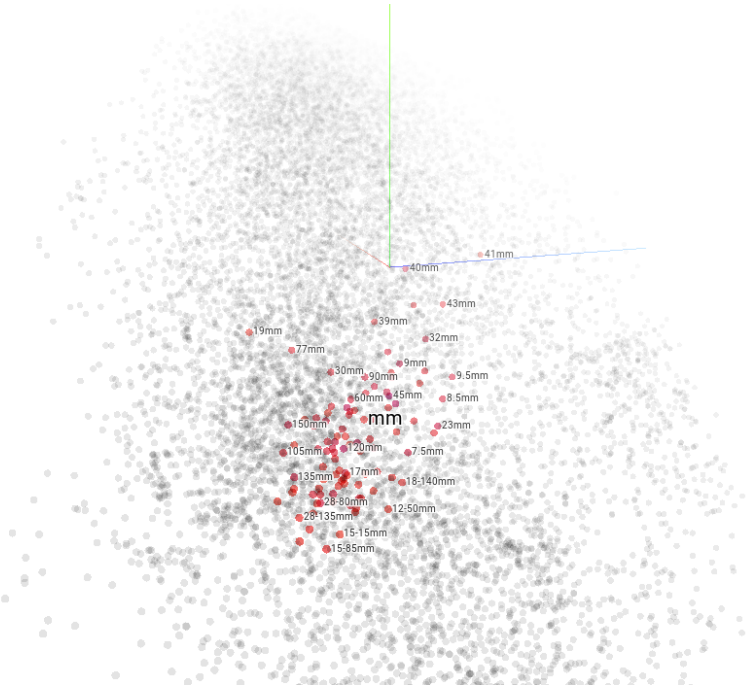


Figure 5.12: Three-dimensional euclidean projection for tokens related to *mm*

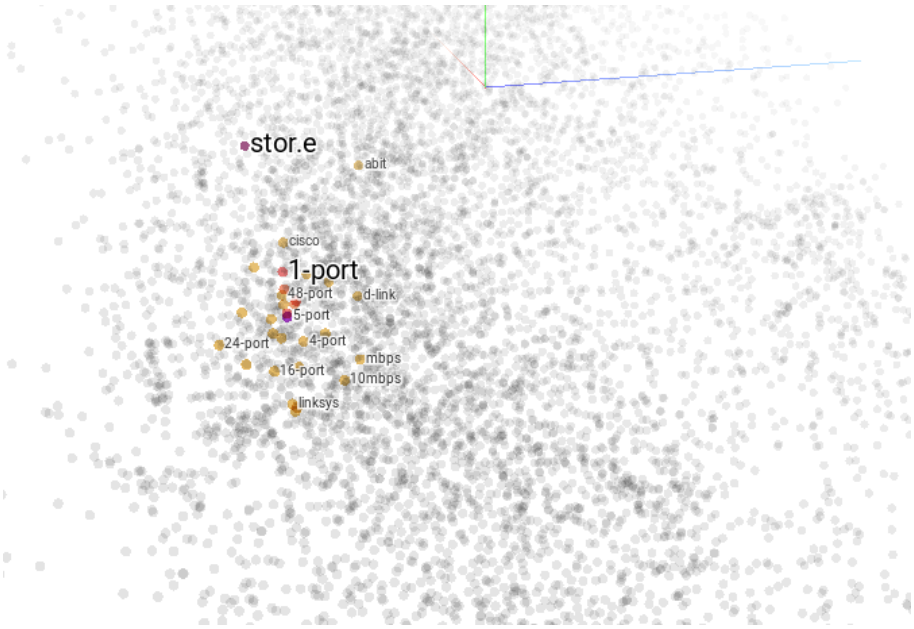


Figure 5.13: Three-dimensional euclidean projection for tokens related to *-port*

Table 5.9: Tables of token counts for tokens with sub-strings 1) *gb*, 2) *mm*, and 3) *-port*.

Token	**Count**	**Token**	**Count**	**Token**	**Count**
2gb	1700	28-80mm	118	1-port	1
3gb	415	18-140mm	26	4-port	112
12gb	237	105mm	242	5-port	15
64gb	1911	8.5mm	13	16-ports	14
180gb	34	28-135mm	41	24-ports	2
960gb	18	15-15mm	2	48-port	38

Table 5.10: Table of BLEU with various SD models

Model	SD	SD + BiRNN	SD + ATT
Ave.	0.587	0.617	0.622

5.3.2 Effect of Re-scoring

In this sub-section we will discuss the outcome of the re-scoring step within the framework. Particularly, we will display the performance of the original stack decoder *SD* against 1) *SD+BiRNN* and 2) *SD+ATT*. *SD+BiRNN* is the model without the attention mechanism, and simply including the BiRNN for creating the vectorial sequence representation. We expect the performances to range from best to worst in this order $(SD + ATT) \geq (SD + BiRNN) \geq SD$.

Figure 5.14 compares three variations of *SD* model and shows that *SD+ATT* not only outperforms the other models in most sizes of source title set, but also display a more robust trend for lower sizes of source title set in the range of [4,8]. Moreover, in Table 5.10 we can see the marginal improvements from *SD* to *SD+ATT* and substantiate our claim for the order of performances with respect to the different models.

Now, having empirically shown that both variance and mean score of BLEU behave as expected upon transformation by decoder, we want to see if re-scorer could indeed select better titles in the pruned search space as produced by the decoder. To do so, we use re-scorer to select from the original source title set search space and also from the resulting search space produced by the decoder. Experiments were performed for *SD+ATT* and and the results are shown in Table 5.11. The results show a three BLEU points improvement for model *SD+ATT*. Since model *SD+ATT* maintains top 10% of the titles, this evidently points to the effectiveness of the decoding process.

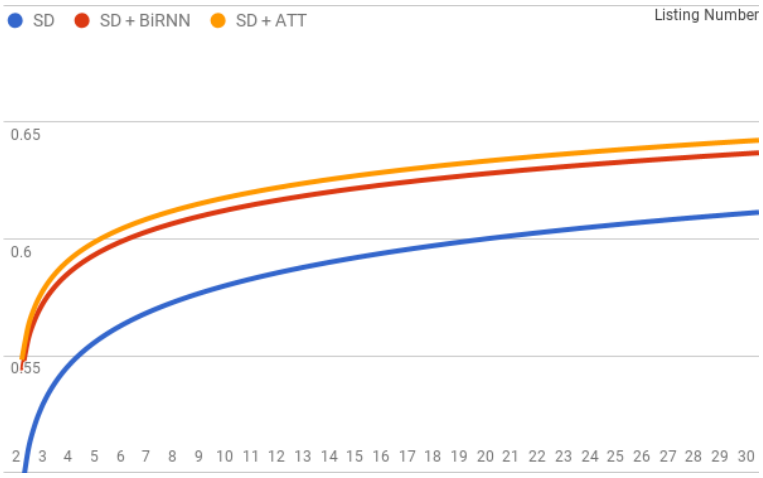


Figure 5.14: Graph of BLEU with variations of *SD*. The plot shows the logarithmic trend lines of models *SD*, *SD + BiRNN*, and *SD + ATT*. It clearly indicates that the performances range from best to worst in this order: $(SD + ATT), (SD + BiRNN), SD$.

Table 5.11: Table of BLEU with and without decoder on *Test-230*

Model	**SD+ATT**	
	w/o decoder	**w/ decoder**
Mean BLEU	0.600	**0.622**

Table 5.12: Mean BLEU of models and the BLEU margin between the models.

	SD+ATT	SD+POS+ATT	Margin
Mean BLEU	0.622	0.591	0.0314

5.3.3 Positional Encoding

Recall in Figure 3.4 we indicate an optional addition of each token's positional encoding. We will now investigate the consequence of this addition and see if it is indeed beneficial to the model performance.

Again, we run a trained re-scorer model including the optional positional encoding by summing it with the original token embedding. We name this model *SD+POS+ATT* and compare it against *SD+ATT* by testing on the test set *Test-230*. The mean BLEU in the source title size of sizes ranging from 2 to 30 is computed as in Table 5.12 while the trend line comparison of the models is depicted in graph 5.15. As such, we see that not only does adding positional encoding hurts the performance, it actually renders it to be below that of the original decoder performance of 0.600. One reason is that information about token position are already included within the feature set. Moreover, by introducing positional encoding we are adding more trainable weights that demand an increase in data size.

5.4 Robustness and Generalizability

In this section, we intend to display the robustness of our framework models against different sizes of source title set. This idea also embodies the general concept of generalizibility where the proposed framework simply has to be trained with a smaller amount of data with small sizes of source title set as it displays a corresponding trend relative to the *Oracle*.

To do so, we first perform experiments with *SD+ATT* against *GSD+ATT* on *Test-230* so as to compare the trend of each models, then we will take the better model to compare against the Oracle. The Oracle is defined as the selected best title using sentence BLEU

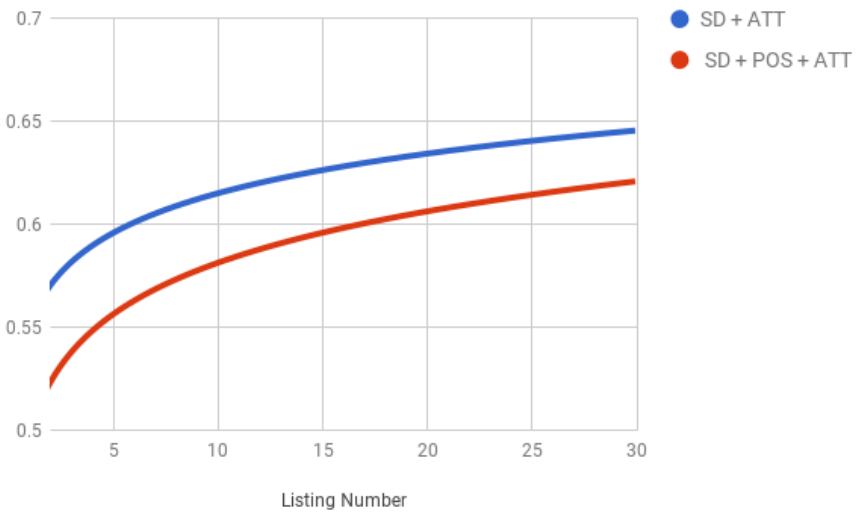


Figure 5.15: Performance comparison between *SD+ATT* against *SD+POS+ATT*. The plot is the logarithmic trend line of each model's performance.

Table 5.13: Performance comparison between *GSD+ATT* and *SD+ATT* against sizes of source title set on *Test-230*

Model	*GSD+ATT*	*SD + ATT*
Mean BLEU	0.661	0.622

with respect to the gold title. As mentioned, we intend to show that our model exhibit a similar trend as the Oracle's BLEU performance.

Figure 5.16 shows a clear competitive edge that *GSD+ATT* has over *SD+ATT* especially for higher sizes of source title set. Average BLEUs for *GSD+ATT* and *SD+ATT* is shown in Table 5.13, showing an approximate improvement of three BLEU points. In fact, the graph displays an increasing margin in BLEU starting from source title set of size 20. Thus, we will employ *GSD+ATT* in order to compare against the Oracle.

As expected, Figure 5.17 exhibits a similar upward trend between the Oracle performance

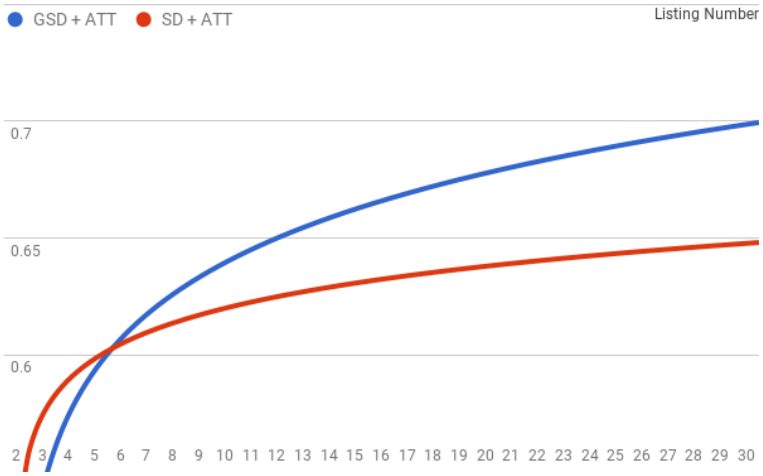


Figure 5.16: Performance comparison between GSD and SD against sizes of source title set on *Test-230*

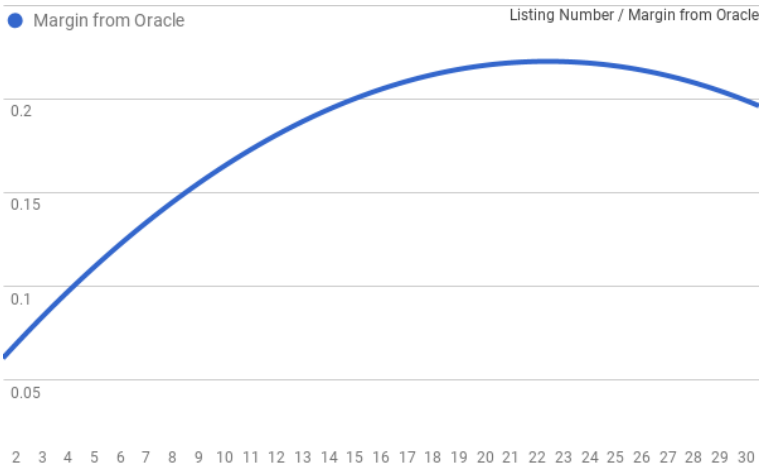


Figure 5.17: Plot of margin of BLEU between GSD+ATT and Oracle against size of source title set on *Test-230*

Table 5.14: Mean BLEU of *Oracle* and *GSD+ATT* against sizes of source title set on *Test-230*

Model	Oracle	GSD+ATT
Mean BLEU	0.833	0.661

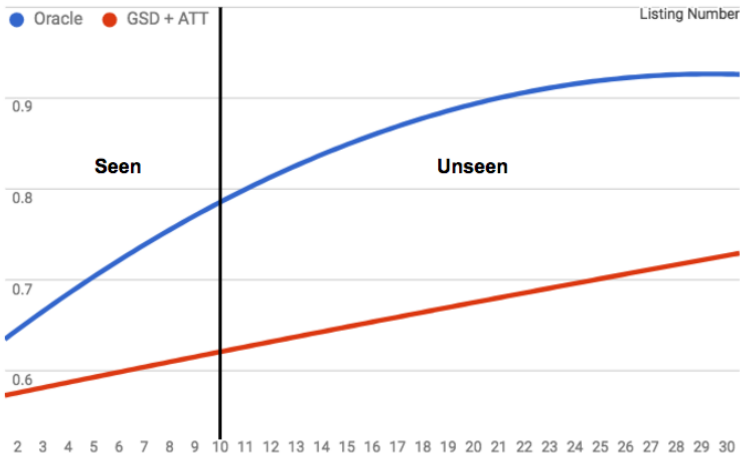


Figure 5.18: Generalizability: A plot of BLEU for both *Oracle* and *GSD+ATT* against source title set size

and that of the *GSD+ATT*, which makes the case for the robustness of the framework against sizes of source title set. We also observe from the Figure 5.17 that after size of source title set increases up to roughly 22, the marginal BLEU starts decreasing. This is achieved having a tiny training set only consisting of data from source title set of sizes in the range [2,10], which is approximately one third of the total testing data. We highlight this finding in Figure 5.18 indicating the seen and unseen range of data. The average performances for Oracle and *GSD+ATT* are shown in Table 5.14. Oracle outperforms model *GSD+ATT* by a margin of one to two BLEU points on *Test-230*.

Table 5.15: Effect of filtering on both models

Model	Unfiltered	Filtered	Diff.
SD+ATT	0.622	**0.626**	0.004
GSD+ATT	**0.661**	0.653	-0.008

5.5 Tractability

Tractability is defined to be whether system could perform within polynomial time. Previously we have shown that the complexity of our decoder algorithm is within such constraint, but we will further attempt to reduce the *possible* exponential increase in time by first filtering the initial source titles.

We intend to do this while ensuring that we do not alter the token distribution. This is achieved via the use of Maximal Marginal Relevance (MMR) which we set to remove half of the set (with α set at 0.5).

5.5.1 Effect of Filtering

Now, we will analyze the effect of MMR on the model performance. After that, we will experimentally show the models (SD and GSD) performances against the effect of filtering. Furthermore, models *SD-ATT* and *GSD-ATT* will be employed as the main focus of this experiment and *Test-210* will be test set of choice.

The goal of this experiment is to attempt to show that the system performs similarly with or without filtering while reducing the running time when the list is filtered. As we see in Table 5.15, both unfiltered and filtered versions of the models produce similar performances with small margins. Similar trend is displayed in Figures 5.19 and 5.20, showing that the effect of filtering actually produces similar performances regardless of the size of source title set.

We also experimented on source title sets of size 1000 with model *SD+ATT*. It was carried

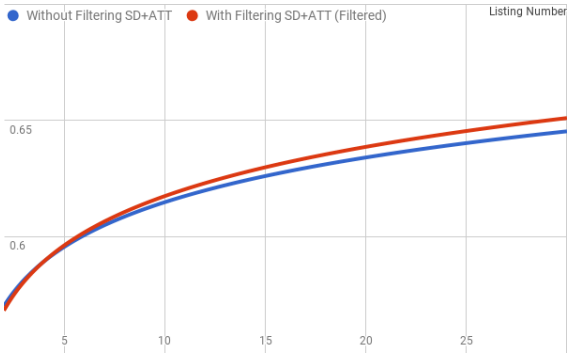


Figure 5.19: Filtering on *SD+ATT*

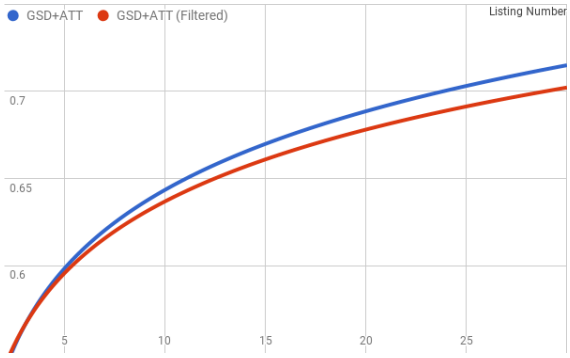


Figure 5.20: Filtering on *GSD+ATT*

Table 5.16: Effect of filtering size on computation time.

Source Title Set Size	**Ratios of comparison**
2	1
5	1
10	1.08
15	1.21
20	1.39
(Original) 1000	1.05

out by allowing MMR to only select 2 titles, and then gradually increase the size of source title set to 20 titles. The results are included in Table 5.16. It is also noticed that even though the computation time increases with filtering, MMR is able to select better initial source title set which results in improvement in BLEU. This finding can be clearly seen in the case of source title set of size 15, where BLEU reaches 0.757 − a significant increase from the original source title set at 0.405. However, we found that MMR adds extra computation time but is capable of producing a slightly better quality pool of initial source titles. As such, we will further test on other source title sets with high set sizes to confirm our findings.

Table 5.17: Complete run on all source title sets with size of ≥ 1000

	Mean BLEU	**Ratios of comparison**
Filtered	0.581	1.5
Unfiltered	0.577	1

To further validate our findings, we perform a complete run on the entire corpus for all source title sets with sizes greater than 1000. We set 1000 to be the threshold for source title set size since we intend to filter out 90% of the titles for set sizes greater than 1000. This would reduce it to be within a more manageable range between source title set size of 100 to 1000 for the original sizes 1000 to 10,000.

We also note down the running time for the program, and the results are recorded in Table 5.17. We see that with 0.04 BLEU points improvement, the running time almost doubles as filtering with MMR is added. We therefore conclude that taking into account the computation time, it is more desirable to not include filtering with MMR.

5.5.2 Computational Cost of Attention Mechanism

We also attempt to measure the computational costs that the model incurs with the inclusion of the attention mechanism. We perform experiments with different sizes of source title set ranging from 5 to 30 at an interval of 5 titles. We compute the percentage increase in computational time upon adding the attention mechanism. As can be seen in Table 5.18, the average running time is about 1.5 times slower. This happens in spite of the increase in size of source title set, showing that the added cost in time is independent of the difference in source titles.

Table 5.18: Execution overhead percentage increase with the addition of attention mechanism.

Source Title Set Size	**Execution Overhead (%)**
5	30.2
10	42.2
15	28.5
20	27.6
25	24.9
30	25.9
Ave.	22.2

5.6 Multi-lingual Settings

We attempt to empirically establish that our model can function in a multi-lingual setting in the given bi-lingual setting given the pre-trained bi-lingual token embedding. Since our decoder does not adopt any language-specific features, we keep the decoder constant, and re-train an additional re-scorer only on the German source titles. We then measure the performances of the re-scorer trained on English source titles (marked with (en)) against that of re-scorer trained only on German source titles (marked with (de)) by obtaining their BLEU on the German data.

Due to the lack of sufficient annotated German data, we have heterogeneous distribution of source title sets based on their set sizes. This could possibly have statistical influence when the source title set count is low, but in general we obtain a meaningful trend between the model performances.

The results are shown in Table 5.19. We see that even though (en)$SD{+}ATT$ was trained entirely on the English data, it performs competitively against (de)$SD{+}ATT$ on the German data. There is an average a 0.027 BLEU difference. We also observe that without the re-

Table 5.19: Re-scorer performance on bi-lingual data

Source Title Set Size	**(en) GSD**	**(de) GSD**	**(en) GSD+ATT**	**(de) GSD+ATT**
2	0.505	0.505	0.533	0.536
3	0.562	0.562	0.572	0.579
4	0.556	0.556	0.579	0.602
5	0.607	0.607	0.64	0.63
6	0.576	0.576	0.6	0.603
7	0.567	0.567	0.587	0.615
8	0.625	0.625	0.63	0.648
9	0.617	0.617	0.634	0.66
10	0.584	0.584	0.601	0.641
Mean BLEU	**0.578**	**0.578**	0.597	**0.613**

scorer, the supposedly multi-lingual decoder performs relatively poor compare to when re-scorer is added. There is in fact a six BLEU points improvement upon adding the English-trained re-scorer when being tested on the German data.

5.7 Baseline Comparison

Finally, we want to compare our models against the original baseline, which adopted the log-linear model. The dataset used in this experiment is the *Test-random*, which consists of 400 randomly selected source title sets. To achieve that, we ran both *SD* and *GSD* against the baseline approach, each with three variations − 1) original decoder, 2) decoder with Bi-RNN, and 3) decoder with Bi-RNN and attention mechanism. The results are shown in Table 5.20, which shows that *GSD* is outperforming *SD* which in turns outperforms the baseline.

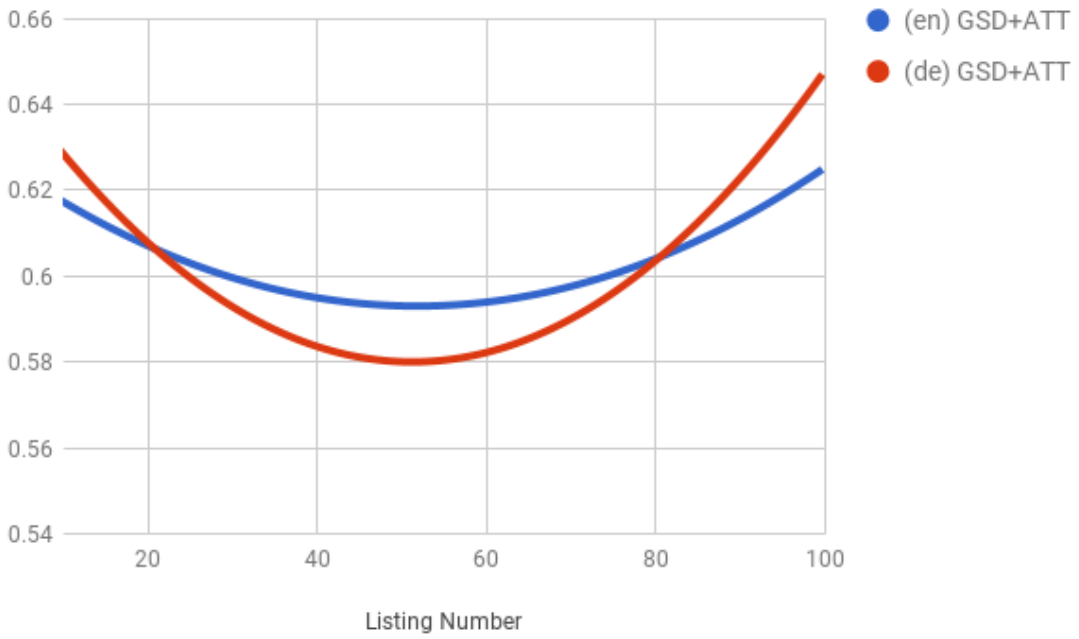


Figure 5.21: Performance comparison between English-trained re-scorer against German-trained re-scorer on German test data

Table 5.20: Baseline comparison

		SD			GSD		
Model	Baseline	-	BiRNN	ATT	-	BiRNN	ATT
Mean BLEU	0.596	0.587	0.617	**0.622**	0.596	0.641	**0.651**

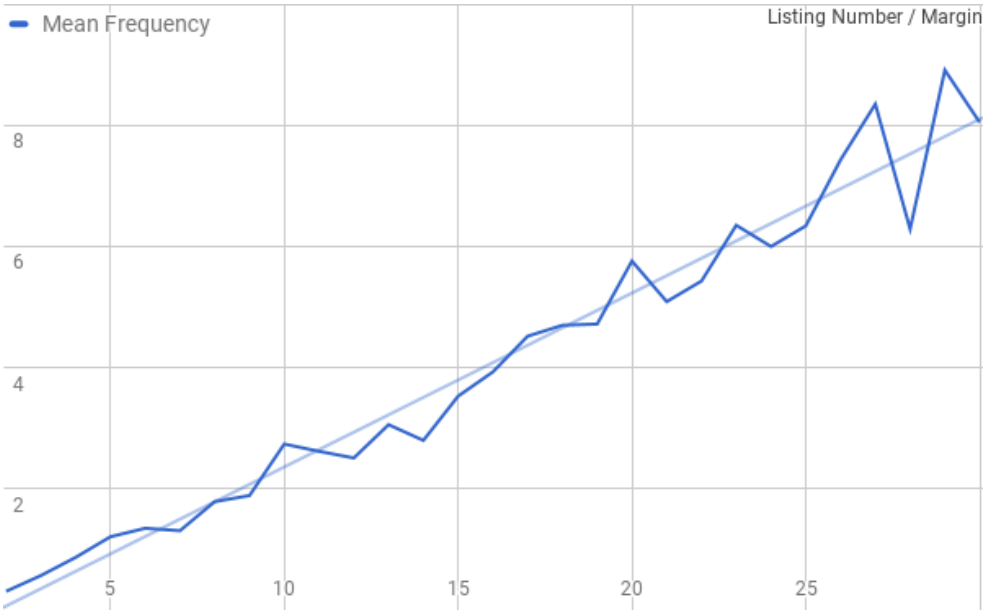


Figure 5.22: Plot of mean count of references appearing in the source title set, averaging over all source title sets with the same source title size. Data is collected from testing set *Test-230*

5.8 Benchmark Comparison

5.8.1 Benchmark-I: MostFreq

Moreover, we observed that reference titles are frequently present in the source title set and that the most commonly appeared titles often end up being the reference title itself. We collect a statistic computing the average frequency of references appearing in the source title set. Figure 5.22 shows the upward trend for counts of reference against the size of source title set. Essentially this is the evidence that as the size of source title set grows, the likelihood of finding the reference within the source title set rises. For this reason, we proposed to simply select the most frequently occurring title in each source title set to be the reference title. This becomes the first benchmark.

Given the above information, it is expected for model *MostFreq* to have an upward

Table 5.21: Table of mean BLEU for models *SD+ATT*, *GSD+ATT*, *MostFreq*, and *MostFreq+OpenNMT*. Test is performed on dataset *Test-230*.

Model	MostFreq	MostFreq+OpenNMT	**GSD+ATT**	SD+ATT
Mean BLEU	0.601	0.392	**0.661**	0.622

trend in terms of performance as well. As such, we compare performances (mean BLEU) of models *SD+ATT*, *GSD+ATT*, and *MostFreq* in Figure 5.23. Not surprisingly, *MostFreq*'s performance improves as the size of source title set increases. The plot also shows substantial margin that our best model *GSD+ATT* has over that of *MostFreq*, indicating that our proposed model is outperforming the benchmark. *SD+ATT* turns out to be slightly less promising at high sizes of source title set. We indicate the mean BLEU over the range of the entire source title set sizes in Table 5.21.

5.8.2 Benchmark-II: MostFreq+OpenNMT

OpenNMT [48] is a state-of-the-art open-source toolkit for Neural Machine Translation (NMT). It includes the use of a structured attention networks and aims to be efficient, extensible, and modular. We trained the toolkit on our training set *Train-210*. To do so, we re-generate from the training set (3,600 source title sets) by matching each source title to the reference title of that same source title set. This comes out to be $2,160$ data points and we then divide it in the 80:20 ratio as training and validation sets, respectively. As a results, it reports a vocabulary size of $22,804$ source-side (source titles) vocabulary, and a mere $6,370$ for the target-side (reference titles).

OpenNMT produces a target title given a source title from the source title set. As such, it is imperative to either 1) select a title to be fed into the trained model, or 2) allow all "translated" output titles to undergo system combination to form a title. We opted for the former to be our approach. The pre-selection of the source title is done via *MostFreq*, which

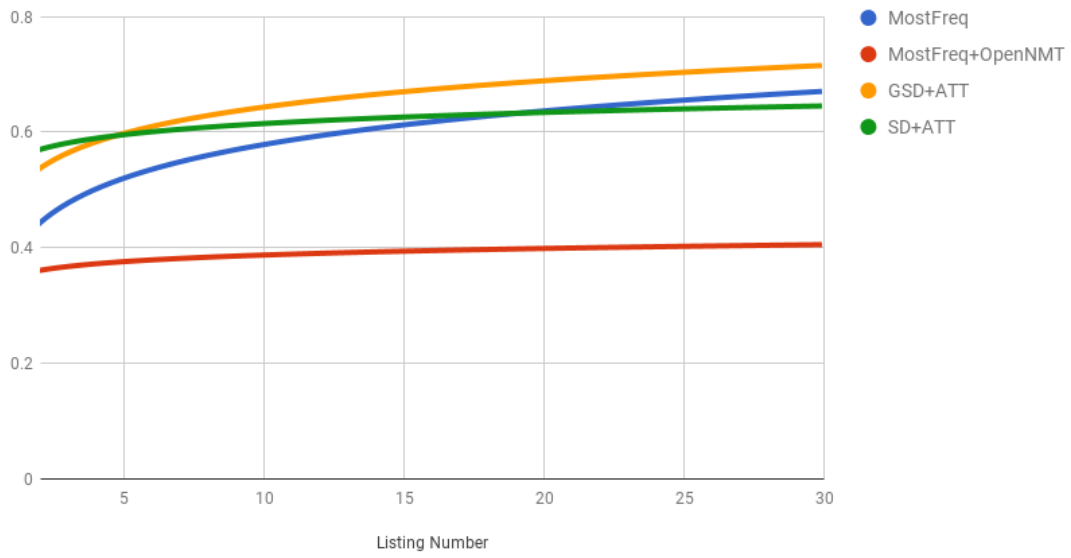


Figure 5.23: Performance comparison between models *SD+ATT*, *GSD+ATT*, *MostFreq*, and *MostFreq+OpenNMT*. Test is performed on dataset *Test-230*.

also makes empirical sense based on our observation as shown in Figure 5.22.

As a result, combination of *MostFreq* and *OpenNMT* yields a very poor performance, worse than if we were to simply select the most occurring title within each source title set as in the case of model *MostFreq*. We attribute the poor performance of *MostFreq+OpenNMT* to the high OOV counts going from source title set to another set. The model is highly reliant on repeated use of tokens for memory transfer so that it could make better judgment generating each tokens. The results show convincing evidence of the superiority of our model over state-of-the-art approaches in neural machine translation. Since the model is representative of the current best neural end-to-end models, we can safely say that our proposed models outperform them by a large margin (0.269 BLEU points).

5.9 Further Experiments and Analysis

After performed analysis on both decoder and re-scorer, we wish to answer a few important theoretical questions we have in mind. These questions represent the last pieces of the jigsaw that will complete our analysis:

- How often and by what amount does the reference titles appear in the source title set?
- How would the models perform with respect to the Oracle if we were to remove the reference titles (human-annotated titles) from the original source title set?
- How would the models and the *Oracle* perform if we were to add the decoder-generated titles to the original source title set and use re-scorer to select from the expanded source title set?

Intuitively, we expect the gold-removed source title set to yield a lower BLEU for Oracle and a higher BLEU for the decoder at lower sizes of source title set. However, we foresee this to change as the size of source title set increases, since this now allows a higher chance of selecting titles that are close to the reference for the case of Oracle. On the other hand, we anticipate the model performance to greatly improve when the reference titles are present in high percentage within the source title set.

Before we carry out these experiments, we compute the mean gold-removed sizes of source title set over 100 products for each set sizes. That is to say, for each size of source title set, we averaged the resulting size of source title set after removing the gold titles over 100 products with the same original size of source title set. The statistics for *Test-230* are shown in Figure 5.24. The mean number of gold titles present in source title set of sizes ranging from 2 to 30 is roughly 4 titles per source title set.

Next, we carry out experiments where models *GSD+ATT*, *SD+ATT*, and the *Oracle* are run on the gold-removed source title set for *Test-230*. Since there are some source title set that contain only the gold titles, we allow all the titles to be retained. Results are shown

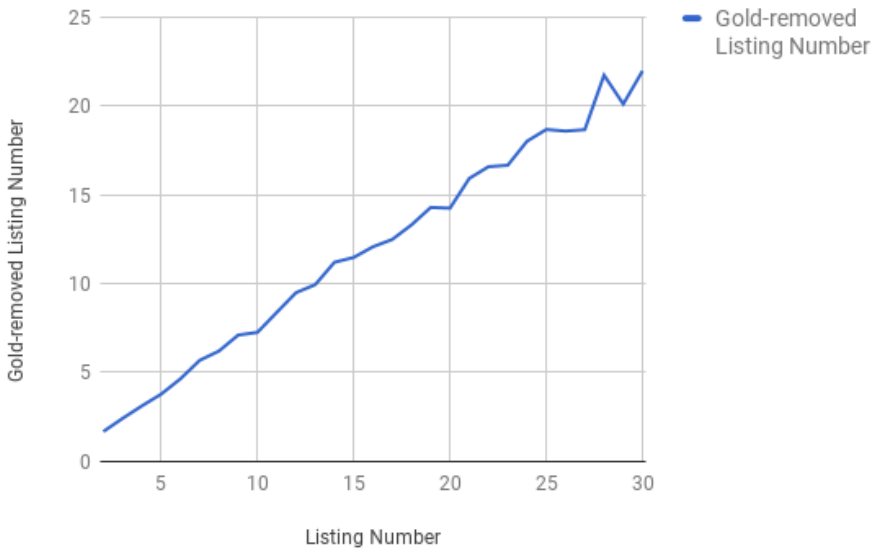


Figure 5.24: Plot of size of source title set against gold-removed source title size. The graph shows a direct proportionality between the original and that of the gold-removed source title set size.

on Figure 5.25. As indicated, there is clearly a positive and linear relationship between the BLEU margin and size of source title set. This goes to say that as more titles are present, it is more likely to find titles that contain a greater degree of resemblances to the reference title.

Recall from our earlier findings that most source title sets have low sizes. Although the data does not favor the proposed models (with respect to the *Oracle*) at high sizes of source title set, it shows promising results in majority of the dataset which contains low sizes of source title set. In fact, the mean BLEU margin is a mere 0.9 BLEU point between source title set size from 2 to 30 and 0.73 between source title set size from 2 to 20. This tiny BLEU margin provides empirical proof that under the worse kind of situation, our proposed models could perform competitively even against the theoretical best.

Effect of adding generated titles to the original source title set also warrants our investi-

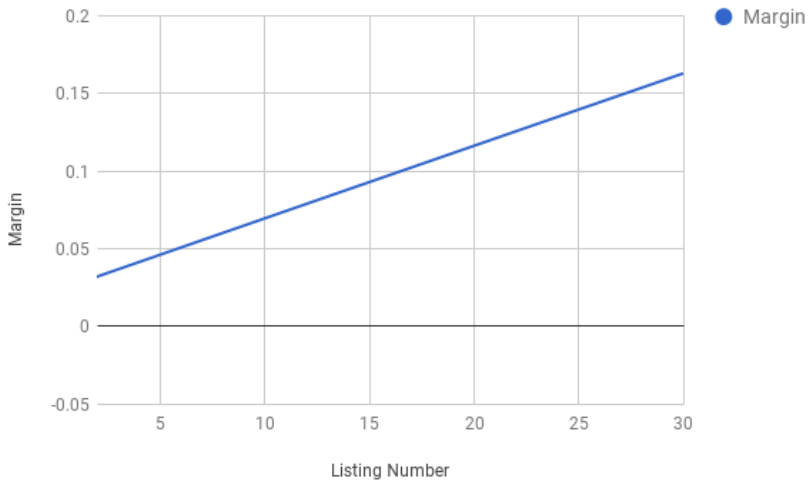


Figure 5.25: Polynomial trend line of margin between *Oracle* BLEU and *GSD+ATT* against size of source title set.

gation. As the precursor to this idea, we have experimented with the effect of pure re-scorer selection from the original set. Importantly, this justified the need for the use of decoder as title generator. Thus right now we will append the decoder-generated titles to the original source title set, and use the re-scorer to select from the new set.

We include the performances of different models in terms of the mean BLEU over different sizes of source title set. The results are shown in Table 5.22, and the trend line is displayed in Figure 5.26. Our results gives an overwhelming evidence that it is preferable for the re-scorer to only select from among the generated titles. Moreover, it also proves once again that our proposed variance-reducing and BLEU-boosting framework is vital to TG.

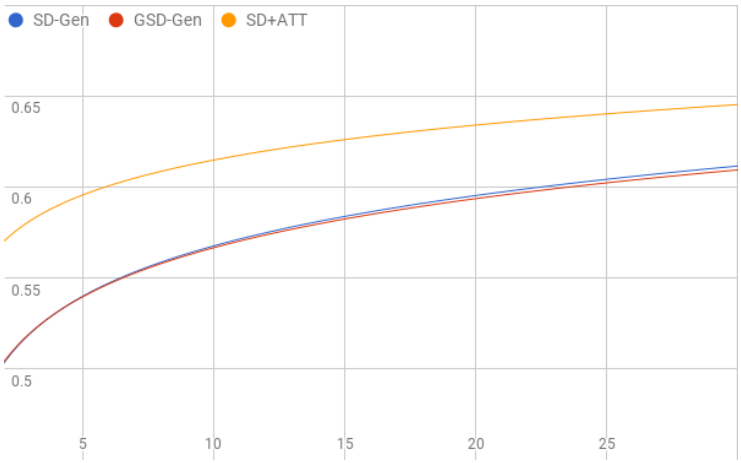


Figure 5.26: Trend line of three models including 1) selection of expanded source title set based on *GSD* decoder, 2) selection of expanded source title set based on *SD* decoder, and 3) Model *SD+ATT* where re-scorer simply select from the generated titles.

Table 5.22: Performances of different approaches. *GSD-Gen* means that the source title set is expanded with titles generated by *GSD* as the decoder, and re-scorer is subsequently used to select from the expanded source title set. *SD+ATT* would be the proposed framework consisting of a decoder, and a re-scorer. The dataset is *Test-230*.

Model		**Mean BLEU**
Selection	**GSD-Gen**	0.577
	SD-Gen	0.579
SD+ATT		**0.622**

5.10 Generation and Analysis

This section records some examples of title generation and discuss the effectiveness of the proposed framework in terms of the generations. More concretely, we details our categorization of generation along with the percentage distribution. Generations are categorized parametrically in two different dimensions − 1) comparison with the Oracle selection (Oracle selects the title with highest BLEU score from the source title set), and 2) whether the generation is identical to that of the gold reference. Table 5.23 denotes the percentages for each categories from (a)-(f). The table shows that the Oracle is still outperforming the system generation in most cases (48%), and the second highest category is the case where system generation is identical to that of the Oracle selection (38%). Surprisingly, our proposed model still managed to outperform Oracle in 14% of the time, meaning that in these cases it either is able to re-generate titles identical to the gold title that is already present in the source title set (c), or is able to generate new titles that was not present anywhere within the source title set (d). In the following subsections, we will delve into (a)-(d) of Table 5.23 in detail by showing generations in the first two major categories of Cat-A.

5.10.1 When the Model is as Good: GSD==Oracle

There exists cases when system generation obtained the same exact score as that as the *Oracle* selection. This can happen under two specific scenarios including 1) outputs from both approaches are identical to the gold title, which also implies that gold titles exist within the source title set, or 2) output differs from the gold title and the system is able to generate and select the next best title from the source title set.

As indicated in Table 5.23, majority of the times when the scores are identical, gold titles are included within the source title set (27% as opposed to 11%). To give a more concrete picture, we will examine an actual set and generations from different approaches. The full set (with source title set of size 10) is provided in Table 5.24, and generations are displayed in Table 5.25. Model *MostFreq* is able to select the gold title given that no other

Table 5.23: Summary of generation types comparing generation by model *GSD+ATT* with Oracle with respect to gold (reference) titles for all source title set with size 10 in *Test-230*. Model *GSD+ATT* is shortened to GSD. Cat-A, or category A, denotes types comparing GSD with Oracle while cat-B, or category B, indicates that it is identical to the Gold title.

Cat-A	**Cat-B**	**Percentage (%)**
1. GSD == Oracle (38%)	**(a) GSD == Gold**	27
	(b) GSD != Gold	11
2. GSD >Oracle (14%)	**(c) GSD == Gold**	5
	(d) GSD != Gold	9
3. GSD <Oracle (48%)	**(e) Oracle == Gold**	19
	(f) Oracle != Gold	29

Table 5.24: Sample source title set with size of 10. Titles identical to the reference titles are in bold.

	Source Titles
1	**Peavey ValveKing 112 50 watt Guitar Amp**
2	Peavey ValveKing 112 W/Footswitch
3	Peavey ValveKing VK112 All-Tube Guitar Amp
4	Peavey ValveKing VK112 All-Tube Guitar Amp
5	**Peavey ValveKing 112 50 watt Guitar Amp**
6	Peavey ValveKing 112 Combo Guitar Amp 50 watt with Foot switch SEE DESCRIPTION
7	Peavey Valveking 112
8	Peavey ValveKing 112 W/Footswitch
9	**Peavey ValveKing 112 50 watt Guitar Amp**
10	"Peavey ValveKing VK112 Tube Amp Combo 1X12" "Good Condition! Sounds Great!!"

Table 5.25: Sample outputs including the gold title, random selection, and outputs of models *GSD+ATT*, *MostFreq*, *MostFreq+OpenNMT* and *SD+ATT*.

Gold Title
Peavey ValveKing 112 50 watt Guitar Amp
Random Selection
"Peavey ValveKing VK112 Tube Amp Combo 1X12" "Good Condition! Sounds Great!!"
MostFreq
Peavey Valveking 112 50 Watt Guitar Amp
MostFreq+OpenNMT
Peavey Windsor 5 15 watt Guitar Amp
GSD+ATT
Peavey Valveking 112 50 Watt Guitar Amp
SD+ATT
Peavey Valveking 112 50 Watt Guitar Amp

titles beat it in frequency of occurrence. Source title set of size 3 that are identical to the gold title. Random selection produces the worst title of it all while both *GSD+ATT* and *SD+ATT* are able to re-create gold-identical titles. Cases as such occur 27 times out of 100, which is almost 1/3 of the times. Lastly, model *MostFreq+OpenNMT* introduces new tokens such as *Windsor*, *5*, and *15* that are absent from the source titles. As much resemblance as the *MostFreq+OpenNMT* generation carries, introduction of new tokens would infringe certain copyright laws, let alone providing misinformation. This opposes the case of random selection which selects a biased title that contain phrases such as "Good Conditions! Sounds Great!", but generally correct information about an item.

Table 5.26: Sample source title set of size 10. Reference title is absent in the source title set in this case.

	Source Titlt set
1	Stihl 122cc MS880 MS 880 chainsaw power head 084 088 090
2	"Stihl MS880 Magnum Professional Gas Chainsaw with 47" "bar and chain."
3	Stihl MS880 MS 880 Magnum Powerhead Only- LOW LOW HOURS - Test Runs Only
4	"Stihl MS 880 Chainsaw 36"" bar"
5	"Stihl MS880 Magnum Professional Gas Chainsaw with 47" bar and chain."
6	PILTZ Stihl MS880 Customised CHAINSAW 72 inch bar and chain
7	STIHL MS880 CHAINSAW POWERHEAD 41 BAR 404CHAIN WRAP HANDLEBAR MS 880 088 084 090
8	"Stihl MS 880 Chainsaw 36"" bar"
9	"USED STIHL CHAINSAW MODEL MS880 25" BAR"
10	STIHL MS880 CHAINSAW POWERHEAD HALF WRAP HANDLEBAR MS 880 088 084 090

5.10.2 The Case of Out-performance: GSD >Oracle

At times, the system outputs better titles than the *Oracle* (14%). This proves that it is possible to find within the bi-gram forest paths that turn out to be superior than the original titles within the source title set. When *GSD+ATT* or *SD+ATT* outperforms the Oracle, it could also indicate the absence of gold titles from the source title set. Another perspective is that quality of source titles are poor enough so that selection method are bound to be inferior. This echoes with our earlier experimental endeavors in proving that our proposed model work nicely with poor quality data.

In this light, it is crucial to dissect the model's successes in terms of the title generation with respect to that of benchmark. Table 5.26 displays a source title set consisting of ten titles (set size of 10) that is related to a specific brand and type of chainsaw. This is an example of a rather noisy set where large variation of titles length is present. There are also some user-specific phrases or item information such as "Test Runs Only", "PILTZ" which makes it harder for the system to make correct choices between the tokens.

Table 5.27: Sample outputs including the gold title, random selection, and outputs of models *GSD+ATT*, *MostFreq*, *MostFreq+OpenNMT* and *SD+ATT*.

Gold Title

Stihl Magnum Ms 880 Chainsaw

Random Selection

Stihl Ms880 Magnum Professional Gas Chainsaw With 47” Bar And Chain.

MostFreq

”Stihl MS 880 Chainsaw 36”” bar”

MostFreq+OpenNMT

”Stihl MS 880 Chainsaw 36”” bar”

GSD+ATT

Stihl Ms880 Ms 880 Chainsaw

SD+ATT

Stihl Ms880 Ms 880 088 090

Sample title generations as shown in Table 5.27 shows a complete mismatch between all outputs and the gold title. Random selection yields an elongated title with additional information such as ”With 47”...”. *MostFreq* and *MostFreq+OpenNMT* produce relatively identical titles to the gold but include specific information regarding certain chainsaws *bar*. *SD+ATT* is a complete failure since it does not state the most fundamental information about the item − the chainsaw. On the bright side, *GSD+ATT* provides a case of success where it was able to generate a title that is not present within the source title set but differs by only a single token. This substantiates our claim that the model is able to generate novel, high-quality titles that otherwise could not be achieved with selection or other benchmarks.

Table 5.28: Token-wise Hypothesis Expansion for model *GSD+ATT*

1-token	**stihl -0.2231435513142097**
	piltz -2.3025850929940455
	used -2.3025850929940455
2-token	**stihl ms880 -0.7339691750802004**
	stihl ms -1.83258146374831
	stihl 122cc -2.525728644308255
	stihl chainsaw -2.525728644308255
3-token	**stihl ms880 ms -2.120263536200091**
	stihl ms880 magnum -2.120263536200091
	stihl ms880 chainsaw -2.120263536200091
	stihl ms880 customised -2.8134107167600364
	stihl ms880 25” -2.8134107167600364
4-token	**stihl ms880 ms 880 -2.120263536200091**
	stihl ms880 magnum professional -2.5257286443082556
	stihl ms880 magnum powerhead -3.218875824868201
	stihl ms880 chainsaw with -3.624340932976365
	stihl ms880 chainsaw 36” -3.624340932976365
	stihl ms880 chainsaw powerhead -3.624340932976365
	stihl ms880 chainsaw power -4.31748811353631
	stihl ms880 chainsaw 72 -4.31748811353631
	stihl ms880 chainsaw model -4.31748811353631
5-token	**stihl ms880 ms 880 chainsaw -2.8134107167600364**
	stihl ms880 ms 880 088 -3.218875824868201
	stihl ms880 ms 880 magnum -3.912023005428146

Token-wise Hypothesis Expansion In Table 5.28, we display the process of hypothesis expansion with model **GSD+ATT** in the worst case scenario, where − 1) gold title does not exist in the source title set, 2) no titles in the set ends with the same tail token *chainsaw*. Beginning with one token, each initial token assigned a score with the State Transition Estimator (STE) function. We observed that at 3-token, there is a tie of scores between *ms*, *magnum*, and *chainsaw*. Since the gold title began with *Stihl Magnum*, there is no attainable way to generate the gold title at this point. However, our system still managed to generate a title ending identically to the Gold title, despite the fact that no title within the source title set has the correct ending token *chainsaw*.

5.10.3 Conclusion

To conclude, we have presented our experiments and findings by first discussing our preliminary findings for determining some of our heuristic decisions including the definition of average token position, minimum title length for filtering, and the beam search threshold value. We then go into exploring the effectiveness of our decoder algorithm and State Transition Estimator (STE) function that was trained non-linearly. To be precise, we attempt to empirically establish the necessity for the decoder in boosting the mean BLEU of title search space while reducing their variance of BLEU at the same time. This allows us to make the claim that the framework does indeed require stack decoder as the precursor to our selection mechanism − the re-scorer.

After which, we provided a visualization of three-dimensional clustering with sub-word token embedding that was trained in an unsupervised manner. We found that the sub-word token embedding is effective in giving tokens or phrases with misspelling or the same unit such as *gb* vectorial representation. Next we further examined the effect of adding re-scorer to the pipeline. We looked at the effect of the system from the perspective of robustness, generalizability, system tractability and the ability to perform competitively in a bi-lingual setting. Our re-scorer model was found to be helpful in boosting the overall performance given different conditions. Additionally, we experimented with possible refinement such as

positional encoding and found that it does not improve our system. Next, we performed numerous more experiments comparing our system with both baseline and benchmark systems.

At the end of the chapter, we carried out several additional experiments which helps us to address aspects that are missing from our previous experimentations. This includes the effect of removal of reference titles on the original source title set on the performance, and so on. Last but not the least, we included an analysis of the generation outputs by our best system, and categorizing it according to the theoretical best, *Oracle*. We then showed sample generations for each of the categories we discussed.

Chapter 6

CONCLUSIONS

The need to enhance users' search experience on e-Commerce sites motivates the task of of Title Generation (TG). TG's novelty also requires new approaches that are rather industrially-driven. Though the initial aim of the task is one that is directly applicable to the e-Commerce industry, we have also managed to propose models that could be of interests to other Natural Language Processing (NLP) community including system combination of Machine Translation (MT) and extractive summarization. This thesis proposed a Generative-Discriminative framework consisting of two major components − 1) generative decoder, and a 2) discriminative re-scorer.

In Chapter 1 we introduced the motivation for our work. TG arose from the need to group source titles based on their attributes. While human-created titles are in abundance, there is a need to create target titles that are product-specific by mimicking the human-created reference title. Then we formulate the task mathematically, without loss of generalization. We also stated the scenarios in which our proposed framework will function in. This includes, 1) ability to perform competitively in multi-lingual settings (bi-lingual in our tests), 2) generalizability, 3) robustness, and 4) tractability. At the end of Chapter 1, we introduced the task of system combination as the main domain of comparison, since it carries much resemblance with TG.

To better position our work, we provided the background to tasks similar to TG. Thus in Chapter 2 we gave an overview of the related works which we categorized based on two dimensions − 1) units of combination and 2) frameworks of approach. Units of combination consist of word-level, phrase-level, and sentence-level. Main categories of frameworks include hypothesis selection, re-decoding, and confusion network decoding. Our work happens at

both the word-level and the sentence-level, and is a combination of both hypothesis selection and re-decoding. Our research on related works determined one of the benchmark for our work to be under the re-decoding category in the domain of Neural Machine Translation (NMT). It combines titles at both the word-level and has a component of beam search.

We outlined the proposed Generative-Discriminative framework details in Chapter 3. The Generative-Discriminative framework works in a pipelined manner, and is a loose form of boosting where two different classifiers are trained sequentially. In our case, data distribution for the two classifiers are altered and some data are removed without replacement. The first classifier, namely the decoder, generates a list of titles which is equivalent to the transformed search space, or data distribution. It is skewed to have both higher mean BLEU and lower variance of BLEU. The desirable consequence is that the discriminator, the re-scorer, could select from this pruned higher-quality of titles. We also discussed in details about the algorithm used for decoder, which is based on stack decoding with beam search. We proposed to use a non-linear approximation for the State Transition Estimator (STE) which we used for hypothesis expansion during the process of stack decoding. STE uses a limited, language-independent feature set so as to allow the model to generalize better. Second component of the framework comes in the form of an attention-based Recurrent Neural Networks (RNN) which takes as input sequence state encodings. Re-scorer approximates the function Sequence Quality Estimator (SQE) which acts as a quality estimator by predicting a BLEU score.

We then delved into the details of the given dataset in Chapter 4. It consists of both English and German annotated data, and amounts to roughly $80,000$ data points. We highlighted some pre-existing trends in the data such as the relationship between source title set sizes and counts of occurrence. Source title sets with smaller sizes occur more frequently in our dataset. In larger sizes of source title set, the probability of finding title with high resemblance to the reference increases.

Lastly, we presented our experiments, results, and findings in Chapter 5. Preliminary findings showed that non-linear function approximation outperformed linear model at all sizes of data, as shown in our learning curve. Next, it was discovered that beam search threshold

value has a positive relationship with decoder's performance. Also, as we had expected, higher sizes of source title set correspond to better generated titles. Our final preliminary exploration provided us with better heuristic decisions that happens in our stack decoding algorithm. This includes average token position determination and the heuristic filtering of titles by length.

For our decoder, we also computed feature importance scores for all the features and determined that Inverse-Document Frequency (IDF) and the average token position are the two most important features. Next, we also established empirically the benefits of the proposed Generative-Discriminative framework. Decoder has the consequence of producing a similar group of higher quality titles. Hypothesis selection in this refined search space is an improvement from the previous approaches. Moreover, decoder is trained on a small dataset with limited sizes of source title set and performs consistently well in much larger, unseen ranges of source title size. This provides strong empirical evidence that the it is robust. This is especially useful for e-Commerce companies that are able to get their hands on limited amount of annotated data but have a urgent demand for large unannotated source title data. Our decoder also limits the need for intensive feature engineering so as to prevent over-fitting and limit language-dependence. Moreover, it is based on the idea of stack decoding allowing the model to generate titles within polynomial time.

Our re-scorer is one that requires the unsupervised pre-training of multi-lingual sub-word token embedding. It was shown to greatly boost the decoder's performance by utilizing token's distribution information and having the attention layer where individual tokens are weighed by their normalized association scores. The result is a promising one since the neural model takes in the same amount of data as that of decoder, and performs competitively in different source title set conditions. This shows its tractability and robustness. Moreover, our experiments also showed that it is able to perform competitively in both German and English data, indicating its potential to work in a multi-lingual setting. Though we would refrain from the claim of multi-linguality, this certainly sheds light on how our system could be adapted to work in different languages.

As far as we know, we have a novel model for re-scorer that works promisingly in an industrially-driven environment. Although the model requires pre-training of token embedding, which is less desirable in the industry, it is of research interests to see how this could be adapted as we do some pre-selection of title data so as to remove the Out-Of-Vocabulary (OOV) words. In the next chapter we will discuss our preliminary ideas for future works.

Chapter 7

FUTURE WORKS

In this chapter, we will discuss the potential next steps to build a system that could serve as an improvement of our current work. There are several areas still waiting to be worked on. Note that this discussion is under the premise of the same task, where the system takes as input a source title set, and outputs a single title.

Our proposed model could serve as the preliminary step to creating an end-to-end system if a large annotated dataset is available. On the other hand, since our focus is on data-scarce settings we ought to ensure that the model complexity does not gets too large. Our results suggest that our proposed framework still has not performed competitively against the theoretical best as produced by the *Oracle* model. We postulate the main reason to be the inability for the model to penalize tokens that are irrelevant to titles. This suggests two things. First is that the attention mechanism is still not effective enough to weed out all of such tokens. Secondly, there needs to be additional modifications to account for this insufficiency.

As such, we propose to train a separate token-relevance classifier that simply takes as input a token and output its probability to be in the title. The training process is simple, since the reference titles have already given clues as to whether a certain token is relevant, i.e., whether it is present in the title or not. We could then utilize the trained classifier and use the predicted scores as part of our existing framework. It could be in either the decoder's hypothesis expansion process or as a feature within re-scorer's state encoding, or both.

Last but not the least, our Generative-Discriminative framework serves as the precursor to a GAN-like adversarial training where decoder's role as a generator and re-scorer's role as a discriminator will play out in a zero-sum game as described in [32]. As of now, our frame-

work's two components work to the same purpose as maximizing the quality of generated title. However, they could also assume adversarial roles during training to reach the global optimum [32] in the Nash equilibrium [64].

Recent works were proposed on the subject of Generative-Adversarial Networks (GANs) that either seeks to improve the training process [79], or adapting it to domains other than Computer Vision (CV) such as the Neural Machine Translation (NMT) [93]. Our preliminary experiments in the two-player adversarial training has shown competitive performance relative to our current work, but it awaits further refinement to possibly adopt the third classifier (token-relevance classifier) in a triple Generative Adversarial Networks (Triple-GAN) [55], in order to fully overcome the current caveats.

Specifically, a two-player adversarial framework could be extended from the earlier probabilistic formulation 7.1:

$$\Pr(X, Y) \approx \underbrace{SQE(t_p \approx t^* | T^{'}, T)}_{\text{Discriminative}} \times \underbrace{STE(T^{'}, T)}_{\text{Generative}} \tag{7.1}$$

where:

t^* is the human-annotated target title that might possibly not be attainable given source title set, T.

t_p is the attainable target title that the framework generate.

$T^{'}$ is the intermediate outputs from decoder.

This can be formulated as a two-player minimax game [32]:

$$\min_{STE} \max_{SQE} U(STE, SQE) =$$

$$E_{x \sim p(x)}[log(STE(x))] + E_{z \sim p_z(z)}[log(1 - SQE(STE(z)))]$$

where generator $STE(\cdot)$ will attempt to fool discriminator $SQE(\cdot)$, while $SQE(\cdot)$ will learn to discern true distribution x from that of the fake distribution z.

One advantage of this framework is that the trained generative decoder does not require pre-trained token embedding and can thus be used with ease at test time. Additionally, we could also incorporate a token-relevance classifier, C. It can be trained jointly in a three-player zero-sum game as in [55], where $\alpha \in (0, 1)$:

$$\min_{STE} \max_{SQE} U(STE, SQE, C) =$$
$$E_{(x,y)\sim p(x,y)}[log(STE(x, y))] + \alpha E_{(x,y)\sim p_c(x,y)}[log(1 - SQE(STE(z)))]$$
$$+ (1 - \alpha)E_{(x,y)\sim p_{STE}(x,y)}[log(1 - SQE(STE(y, z), y))]$$

In this case, classifier C and generator STE will generate pseudo input-label pairs (x,y), which is then sent to the discriminator SQE for judgment [55]. As such, all three models are trained jointly in a three-player game.

BIBLIOGRAPHY

[1] Nguyen Bach, Qin Gao, Stephan Vogel, and Alex Waibel. Tris: A statistical sentence simplifier with log-linear models and margin-based discriminative training. In *IJCNLP*, pages 474–482, 2011.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, volume 29, pages 65–72, 2005.

[4] Richard Bellman. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences*, 38(8):716–719, 1952.

[5] Richard Berk and John M MacDonald. Overdispersion and poisson regression. *Journal of Quantitative Criminology*, 24(3):269–284, 2008.

[6] Alina Beygelzimer, Varsha Dani, Tom Hayes, John Langford, and Bianca Zadrozny. Error limiting reductions between classification tasks. In *Proceedings of the 22nd international conference on Machine learning*, pages 49–56. ACM, 2005.

[7] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[8] Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336. ACM, 1998.

[9] Boxing Chen and Colin Cherry. A systematic comparison of smoothing techniques for sentence-level bleu. In *WMT@ ACL*, pages 362–367, 2014.

[10] Kuan-Yu Chen, Shih-Hung Liu, Berlin Chen, Hsin-Min Wang, Ea-Ee Jan, Wen-Lian Hsu, and Hsin-Hsi Chen. Extractive broadcast news summarization leveraging recurrent neural network language modeling techniques. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 23(8):1322–1334, 2015.

[11] Wenhu Chen, Evgeny Matusov, Shahram Khadivi, and Jan-Thorsten Peter. Guided alignment training for topic-aware neural machine translation. *arXiv preprint arXiv:1607.01628*, 2016.

[12] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016.

[13] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*, 2016.

[14] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[15] Sumit Chopra, Michael Auli, Alexander M Rush, and SEAS Harvard. Abstractive sentence summarization with attentive recurrent neural networks. *Proceedings of NAACL-HLT16*, pages 93–98, 2016.

[16] Ms Pooja S Choudhari and SS Nandgaonkar. A survey on short text summarization of comment streams on social network sites.

[17] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[18] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[19] Jacob Devlin and Spyros Matsoukas. Trait-based hypothesis selection for machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 528–532, Montréal, Canada, June 2012. Association for Computational Linguistics.

[20] Jacob Devlin, Antti-Veikko I Rosti, Sankaranarayanan Ananthakrishnan, and Spyros Matsoukas. System combination using discriminative cross-adaptation. In *IJCNLP*, pages 667–675, 2011.

[21] Jinhua Du and Andy Way. Using TERp to augment the system combination for SMT. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas*, 2010.

[22] Christopher Dyer, Smaranda Muresan, and Philip Resnik. Generalizing word lattice translation. Technical report, DTIC Document, 2008.

[23] Nader Fallah, Hong Gu, Kazem Mohammad, Seyyed Ali Seyyedsalehi, Keramat Nourijelyani, and Mohammad Reza Eshraghian. Nonlinear poisson regression using neural networks: a simulation study. *Neural Computing and Applications*, 18(8):939, 2009.

[24] Yang Feng, Yang Liu, Haitao Mi, Qun Liu, and Yajuan Lü. Lattice-based system combination for statistical machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1105–1113, Singapore, August 2009. Association for Computational Linguistics.

[25] Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

[26] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

[27] Sahar Ghannay and Loïc Barrault. Using hypothesis selection based features for confusion network mt system combination. In *Proceedings of the 3rd Workshop on Hybrid Approaches to Machine Translation (HyTra)*, pages 2–6, Gothenburg, Sweden, April 2014. Association for Computational Linguistics.

[28] Daniel Jacob Gillick. *The elements of automatic summarization.* PhD thesis, UNIVERSITY OF CALIFORNIA, BERKELEY, 2011.

[29] Jesús González-Rubio and Francisco Casacuberta. The upv-prhlt combination system for wmt 2011. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 140–144. Association for Computational Linguistics, 2011.

[30] Jesús González-Rubio and Francisco Casacuberta. Improving the minimum Bayes' risk combination of machine translation systems. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, 2013.

[31] Jesús González-Rubio, Alfons Juan, and Francisco Casacuberta. Minimum bayes-risk system combination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1268–1277. Association for Computational Linguistics, 2011.

[32] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[33] David Graff and C Cieri. English gigaword corpus. *Linguistic Data Consortium*, 2003.

[34] Jiatao Gu, Kyunghyun Cho, and Victor OK Li. Trainable greedy decoding for neural machine translation. *arXiv preprint arXiv:1702.02429*, 2017.

[35] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*, 2016.

[36] Hongyu Guo. Generating text with deep reinforcement learning. *arXiv preprint arXiv:1510.09202*, 2015.

[37] Xiaodong He and Kristina Toutanova. Joint optimization for machine translation system combination. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1202–1211. Association for Computational Linguistics, 2009.

[38] Xiaodong He, Mei Yang, Jianfeng Gao, Patrick Nguyen, and Robert Moore. Indirect-hmm-based hypothesis alignment for combining outputs from machine translation systems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 98–107. Association for Computational Linguistics, 2008.

[39] Kenneth Heafield and Alon Lavie. Voting on n-grams for machine translation system combination. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas*, 2010.

[40] Almut Silja Hildebrand and Stephan Vogel. Combination of machine translation systems via hypothesis selection from combined n-best lists. In *MT at work: Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas*, pages 254–261, 2008.

[41] Almut Silja Hildebrand and Stephan Vogel. Cmu system combination via hypothesis selection for wmt'10. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 307–310, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

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

[43] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Controllable text generation. *arXiv preprint arXiv:1703.00955*, 2017.

[44] Shyamsundar Jayaraman and Alon Lavie. Multi-engine machine translation guided by explicit word matching. In *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, pages 101–104. Association for Computational Linguistics, 2005.

[45] Khosrow Kaikhah. Text summarization using neural networks. 2004.

[46] Damianos Karakos, Jason Eisner, Sanjeev Khudanpur, and Markus Dreyer. Machine translation system combination using itg-based alignments. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 81–84. Association for Computational Linguistics, 2008.

[47] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[48] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*, 2017.

[49] Philipp Koehn. Neural machine translation. *arXiv preprint arXiv:1709.07809*, 2017.

[50] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007.

[51] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[52] Shankar Kumar and William Byrne. Minimum bayes-risk decoding for statistical machine translation. Technical report, DTIC Document, 2004.

[53] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[54] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[55] Chongxuan Li, Kun Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. *arXiv preprint arXiv:1703.02291*, 2017.

[56] Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[57] Hans Peter Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4):309–317, 1957.

[58] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[59] Wei-Yun Ma. *Hybrid System Combination for Machine Translation: An Integration of Phrase-level and Sentence-level Combination Approaches*. Columbia University, 2014.

[60] Wei-Yun Ma and Kathleen McKeown. System combination for machine translation through paraphrasing. In *EMNLP*, pages 1053–1058, 2015.

[61] Wolfgang Macherey and Franz Josef Och. An empirical study on computing consensus translations from multiple machine translation systems. In *EMNLP-CoNLL*, pages 986–995. Citeseer, 2007.

[62] Prashant Mathur, Nicola Ueffing, and Gregor Leusch. Generating titles for millions of browse pages on an e-commerce site.

[63] Evgeny Matusov, Nicola Ueffing, and Hermann Ney. Computing consensus translation for multiple machine translation systems using enhanced hypothesis alignment. In *EACL*, 2006.

[64] John F Nash et al. Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49, 1950.

[65] Ani Nenkova, Kathleen McKeown, et al. Automatic summarization. *Foundations and Trends® in Information Retrieval*, 5(2–3):103–233, 2011.

[66] John Neter, Michael H Kutner, Christopher J Nachtsheim, and William Wasserman. *Applied linear statistical models*, volume 4. Irwin Chicago, 1996.

[67] Andrew Y Ng and Michael I Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in neural information processing systems*, pages 841–848, 2002.

[68] Sergei Nirenburg and Robert Frederking. Toward multi-engine machine translation. In *Proceedings of the workshop on Human Language Technology*, pages 147–151. Association for Computational Linguistics, 1994.

[69] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics, 2003.

[70] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51, 2003.

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

[72] Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*, 2016.

[73] V. Punyakanok and D. Roth. The use of classifiers in sequential inference. In *NIPS*, pages 995–1001. MIT Press, 2001.

[74] Antti-Veikko I Rosti, Necip Fazil Ayan, Bing Xiang, Spyridon Matsoukas, Richard M Schwartz, and Bonnie J Dorr. Combining outputs from multiple machine translation systems. In *HLT-NAACL*, pages 228–235, 2007.

[75] Antti-Veikko I Rosti, Xiaodong He, Damianos Karakos, Gregor Leusch, Yuan Cao, Markus Freitag, Spyros Matsoukas, Hermann Ney, Jason R Smith, and Bing Zhang. Review of hypothesis alignment algorithms for mt system combination via confusion network decoding. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 191–199. Association for Computational Linguistics, 2012.

[76] Antti-Veikko I Rosti, Spyridon Matsoukas, and Richard Schwartz. Improved word-level system combination for machine translation. In *Annual Meeting-Association for Computational Linguistics*, volume 45, page 312, 2007.

[77] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[78] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In Llus Mrquez, Chris Callison-Burch, Jian Su,

Daniele Pighin, and Yuval Marton, editors, *EMNLP*, pages 379–389. The Association for Computational Linguistics, 2015.

[79] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[80] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[81] Khe Chai Sim, William J Byrne, Mark JF Gales, Hichem Sahbi, and Philip C Woodland. Consensus network decoding for statistical machine translation system combination. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, pages IV–105. IEEE, 2007.

[82] Daria Sorokina, Rich Caruana, Mirek Riedewald, and Daniel Fink. Detecting statistical interactions with additive groves of trees. In *Proceedings of the 25th international conference on Machine learning*, pages 1000–1007. ACM, 2008.

[83] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21, 1972.

[84] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[85] Hirokazu Suzuki. Automatic post-editing based on SMT and its selective application by sentence-level automatic quality evaluation. In *Proceedings of the 13th Machine Translation Summit (MT Summit XIII)*, pages 156–163. International Association for Machine Translation, 2011.

[86] Jian Tang, Yifan Yang, Sam Carton, Ming Zhang, and Qiaozhu Mei. Context-aware natural language generation with recurrent neural networks. *arXiv preprint arXiv:1611.09900*, 2016.

[87] Michael Tsang, Dehua Cheng, and Yan Liu. Detecting statistical interactions from neural network weights. *arXiv preprint arXiv:1705.04977*, 2017.

[88] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*, 2016.

[89] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[90] Mengqiu Wang and Christopher D Manning. Effect of non-linear deep architecture in sequence labeling. In *IJCNLP*, pages 1285–1291, 2013.

[91] Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, September 2015.

[92] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *arXiv preprint arXiv:1508.01745*, 2015.

[93] Lijun Wu, Yingce Xia, Li Zhao, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. Adversarial neural machine translation. *arXiv preprint arXiv:1704.06933*, 2017.

[94] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[95] Tian Xia, Zongcheng Ji, Shaodan Zhai, Yidong Chen, Qun Liu, and Shaojun Wang. Improving alignment of system combination by using multi-objective optimization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 535–544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[96] Shasha Xie and Yang Liu. Using confusion networks for speech summarization. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 46–54. Association for Computational Linguistics, 2010.

[97] Daguang Xu, Yuan Cao, and Damianos Karakos. Description of the jhu system combination scheme for wmt 2011. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 171–176. Association for Computational Linguistics, 2011.

[98] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of NAACL-HLT*, pages 1480–1489, 2016.

[99] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[100] Wojciech Zaremba. An empirical exploration of recurrent network architectures. *Journal of Machine Learning Research*, 2015.

[101] Xingxing Zhang and Mirella Lapata. Sentence simplification with deep reinforcement learning. *arXiv preprint arXiv:1703.10931*, 2017.

VITA

Chuan-Yi Chang is a Master student at the University of Washington majoring in Computational Linguistics. He has interests in Natural Language Generation (NLG), Machine Translation (MT), Text Summarization, Semantic Textual Similarity (STS), and Textual Entailment (TE).

He welcomes your comments at `cyc025@uw.edu`.