Learning to Balance Lead Bias in News Summarization

Matt Grenander

Department of Computer Science
McGill University, Montreal

July 8, 2020

A thesis submitted to McGill University in partial fulfilment of the requirements of the degree of Master of Science.

Acknowledgements

First, I would like to thank my supervisors Jackie Chi Kit Cheung and Annie Louis. I am thankful for their guidance in exploring research topics, carrying out practical research and communicating these ideas effectively. They have helped me think critically and become a better researcher.

I also want to thank my colleagues and friends at MILA for their helpful discussions, patience and feedback. In particular, I am grateful to Yue Dong for our many conversations that led to new research directions and collaborations.

Abstract

As technology increases the tremendous amount of text we encounter each day, a clear need has arisen for tools to filter out noise and highlight key phrases. **Automatic summarization** systems have emerged as a viable solution to sort through and intelligently condense text. In this thesis, we focus on **extractive summarization**, in which the system selects text snippets from a source document that best represent the given text. Extractive summarization systems must learn complex lexical representations so that they are able to rank sentences in order of relevance, while minimizing redundancy in the output summary. For news articles, extractive summarization systems have been shown to exhibit a bias towards selecting content from an article's lead sentences, even when these sentences are irrelevant to the overall text (Kedzie, McKeown, and Daume III 2018). We investigate this phenomenon in detail, showing that when an article's sentence order is permuted, state-of-the-art systems drastically underperform. To alleviate this problem, we propose an auxiliary objective function which encourages the model to look beyond a document's leading sentences and properly value each sentence. We show that this auxiliary objective significantly improves summarization performance, particularly in cases where the article's leading sentences constitute a poor summary. We extend this approach to a novel summarization method that classifies documents into two distinct groups: ones in which leading phrases constitute a strong summary, and ones in which they form a poor summary. Two separate systems then summarize the two groups. We show that this approach is promising, though accurate classification remains an obstacle towards improved summarization.

Abrégé

L'usage de la technologie au quotidien nous confrontant à de plus en plus à d'éléments textuels, nous avons plus que jamais besoin d'outils pour filtrer et synthétiser les concepts clés dans un corps de texte. Les **systèmes de résumé automatiques** apparaissent comme des solutions viables pour organiser et condenser du texte intelligemment. Dans cette thèse, nous nous concentrons sur **la synthèse extractive**, dans laquelle le système sélectionne les extraits du document source représentant au mieux l'idée générale du texte. Les systèmes extractifs doivent apprendre des représentations lexicales complexes afin de pouvoir classer les phrases par ordre de pertinence, tout en minimisant leur redondance dans le résumé final. Pour les articles de presse, il a été démontré que les systèmes extractifs présentent un biais car ils sélectionnent d'emblée les premières phrases de l'article, même lorsque ces phrases ne sont pas pertinentes pour résumer le texte global (Kedzie, McKeown, and Daume III 2018). Nous étudions ce phénomène en détail, en démontrant le déclin drastique des performances de technologies de pointe lorsque l'ordre des phrases au sein d'un article est perturbé. Pour atténuer ce problème, nous proposons un objectif auxiliaire qui encourage le modèle à poursuivre son analyse au-delà des premières phrases d'un document, et à évaluer correctement chaque phrase. Nous montrons que cet objectif auxiliaire améliore considérablement la qualité de synthèse, en particulier dans les cas où les premières phrases de l'article constituent un mauvais résumé. Nous étendons cette approche à une nouvelle méthode de synthèse, qui classe les documents en deux groupes distincts : ceux dont les premières phrases constituent un bon résumé, et ceux pour lesquelles elles sont insuffisantes. Des systèmes séparés résument les deux groupes. Nous montrons que cette approche est prometteuse, bien qu'il soit nécessaire d'optimiser davantage le système pour classer les articles avec plus de précision.

Contents

List of Figures

List of Tables

1

Introduction

Natural Language Processing (NLP) is an important subfield of artificial intelligence concerned with designing systems that understand and respond to human language. Complex natural language tasks often require intricate solutions, and many NLP problems remain open research areas. Among these topics, **automatic summarization** aims to create systems able to write concise summaries of various text sources. Automatic summarization's use cases are diverse: these systems could be used to summarize informal text such as emails or extract key points from formal literature such as medical reports or legal documents (Zhang et al. 2018; Kornilova and Eidelman 2019). Professions that require studying large amounts of text, such as doctors, lawyers or analysts, could greatly benefit from needing less time to grasp key concepts. The task's usefulness attracts substantial research interest, and forms the topic of this thesis.

Summarization systems typically follow one of two approaches. **Abstractive** summarizers generate summaries token-by-token, requiring the system to both understand the text in-depth and produce a grammatical summary. On the other hand, **extractive** summarizers create summaries by selecting relevant text spans – usually sentences – from the source document. This thesis will focus on analyzing and improving the extractive approach, particularly in the news domain.

Extractive summarizers have historically focused on selecting the most *relevant* text snippets while reducing *redundancy* between selected segments. This can be accomplished in

Article: Bangladesh beat fellow World Cup quarter-finalists Pakistan by 79 runs in the first one-day international in Dhaka. Tamim Iqbal and Mushfiqur Rahim scored centuries as Bangladesh made 329 for six and Pakistan could only muster 250 in reply. Pakistan will have the chance to level the three-match series on Sunday when the second ODI takes place in Mirpur. Bangladesh elected to bat after winning the toss but struggled to 67 for two in the 20th over after Soumya Sarkar was run out by Wahab Riaz for 20 and Mahmudullah was bowled by Rahat Ali. Tamim and Mushfiqur set about repairing Bangladesh's stuttering innings and did just that by putting on 178 runs in 21.4 overs for the third wicket, during which time Tamim notched his fifth ODI century. He continued to plunder runs with ease but went for one big shot too many against Wahab and was caught at mid-off to leave Bangladesh on 245 for three with nine overs left. His 135-ball knock of 132 included 15 fours and three sixes. Two sixes in the 43rd over took Mushfiqur into the nineties and he joined Iqbal in passing three figures by hitting Saeed Ajmal for successive fours in the 45th. Mushfiqur perished in the 48th over when he edged Wahab behind to depart for a 77-ball innings of 106 which included 13 fours and two sixes. Shakib al Hasan and Sabbir Rahman scored 30 runs in 2.2 overs before falling to Wahab (four for 59) in the final over as the hosts set pakistan 330 for victory. Azhar Ali and Sarfraz Ahmed put on 53 runs for the first wicket before the latter slog-swept Arafat Sunny to deep backward square leg to leave Pakistan one down in the 11th over. Mohammad Hafeez was then run out to leave Pakistan stuttering on 59 for two but there was some respite when Azhar notched his fifth ODI half-century in the 20th over to celebrate his first match as captain in fine style.
Reference: Bangladesh beat fellow World Cup quarter-finalists Pakistan by 79 runs. Tamim Iqbal and Mushfiqur Rahim scored centuries for Bangladesh. Bangladesh made 329 for six and Pakistan could only muster 250 in reply. Pakistan will have the chance to level the three-match series on Sunday.

Table 1.1: Leading sentences often form a strong baseline for news summarization, such as in this example. Here, the highlighted passage indicates the most closely related sentences to the reference summary. Note that the article has been truncated for conciseness.

numerous ways, such as a linear combination of these two features (Carbonell and Goldstein 1998), or more sophisticated formulations using integer linear programming or graphical methods (McDonald 2007; Mihalcea and Tarau 2004). More recently, state-of-the-art summarizers have overwhelmingly been developed using deep learning approaches, and this thesis focuses on deficiencies within these methods.

Different text domains often carry idiosyncratic traits and require different summarization approaches. For example, a system summarizing emails may benefit from using an abstractive approach, since emails are typically comprised of a conversational nature and may not contain relevant summary-worthy text snippets. In the same way, news summarization also features unique characteristics that affect how we summarize these texts. News articles,

Article: Standing up for what you believe. What does it cost you? What do you gain? Memories Pizza in the Indiana town of Walkerton is finding out. The family-run restaurant finds itself at the center of the debate over the state's religious freedom restoration act after its owners said they'd refuse to cater a same-sex couple's wedding. "If a gay couple was to come and they wanted us to bring pizzas to their wedding, we'd have to say no", Crystal O'Connor told CNN affiliate WBND-TV in South Bend. The statement struck at the heart of fears by critics, who said the new law would allow businesses to discriminate against gays and lesbians. They called for boycotts. But supporters also rallied. And by the end of the week, they had donated more than $842,000 for the business. Social media unloaded on the pizzeria in the community of 2,100 people that few folks outside northern Indiana knew existed before this week. riskyliberal tweeted : "Dear #memoriespizza. no. My boycotting your business because I don't like your religious bigotry is not a violation of your freedom to practice your religion." "Don't threaten #memoriespizza" tweeted aanda. "Just mock them for their ignorance." Bad reviews flooded the restaurant's Facebook page, most having little to do with the quality of the food. Many too vulgar to share. "Do you really want to financially support a company that treats some of your fellow citizens like second class citizens? Boycott memories pizza!!" said Rob Katz of Indianapolis. "Let's hope they either rethink their policy or the free market puts them out of business." But one outburst in particular shut down the restaurant Wednesday and was expected to do the same Thursday. "Who's going to Walkerton with me to burn down Memories Pizza?" Jessica Dooley of Goshen tweeted, according to the Walkerton police department. The account has been deleted since the tweet was posted. Detectives who investigated have recommended charges of harassment, intimidation and threats, according to Charles Kulp, assistant police chief.
Reference: Indiana town's Memories Pizza is shut down after online threat. Its owners say they'd refuse to cater a same-sex couple's wedding.

Table 1.2: In some cases, such as this one, models must learn that leading sentences do not always constitute meaningful content. As in Table 1.1, the highlighted passages indicate sentences that reflect the content from the reference summary. The article has been truncated for conciseness.

especially event-based journalism, usually follow an inverted pyramid scheme, in which the main facts are placed near the article's starting point. Using the first three sentences of an article as a summary is often used as a strong baseline (Nenkova 2005), and many systems naturally exploit position cues when extracting a summary (Hong and Nenkova 2014; Schiffman, Nenkova, and McKeown 2002). For example, consider the article in Table 1.1. While the article's first three sentences include slightly extraneous information, overall it closely mimics the reference summary and undoubtedly represents a strong extractive summary.

However, in many situations the leading sentences may not convey the most meaningful summary content. For example, consider the article in Table 1.2. In this case, summarization models should recognize that the preamble does not contribute relevant information and

Model	Lead Overlap (%)	ROUGE-1, -2, -L Average
Oracle	27.24	47.43
NeuSum (Q. Zhou et al. 2018)	58.24	31.52
RNES (Wu and Hu 2018)	68.44	32.57
BanditSum (Dong et al. 2018)	69.87	32.82

Table 1.3: On the CNN / Dailymail dataset (Hermann et al. 2015), recent models frequently select leading sentences far above the rate that an oracle summarizer does. The third column measures a summarization performance metric named ROUGE (higher is better).[1] The striking difference between recent models and the oracle indicates that current models remain far from optimal.

exclude these sections from the output summary. Likewise, another challenge models face is recognizing that important content may occur near the end of the article, as in this example.

Previous work has shown that more than 20-30% summary-worthy sentences come from the second half of news documents (Nallapati, B. Zhou, et al. 2016; Kedzie, McKeown, and Daume III 2018). It is therefore crucial that systems properly balance position cues with semantic representations of the text. Alas, previous studies suggest that most recent neural methods predominantly pick sentences from the lead, and that their content selection performance drops greatly when the position cues are withheld (Kedzie, McKeown, and Daume III 2018). Table 1.3 provides a sample of how often recent systems select sentences from the lead. We include an "oracle" extractive summarizer in which we compute the 3 highest-scoring sentences with respect to ROUGE, a measure of lexical overlap between the system and reference summary.[2] The striking difference between the oracle extractive summarizer and other recent systems indicates that these models' reliance on positional cues is a serious deficiency.

This trend is particularly worrying since it suggests that current systems ignore learning the document's details in favour of exploiting simple positional cues. This learning bottleneck has implications beyond news summarization: many other summarization domains may also hold similar idiosyncrasies that models may be exploiting instead of learning suitable

[1] See Section 2.2.2 for an overview of ROUGE.
[2] See footnote 1.

representations. For this reason, it is important to design methods that promote learning beyond simple cues, and encourage models to learn deeper semantic representations.

1.1 Thesis Outline

In this thesis, we explore to what degree models are affected by positional biases in news summarization, primarily using the recent BanditSum model (Dong et al. 2018) to explore these issues. We then formulate a method to counter these biases using an auxiliary target objective, showing that this technique leads to better summaries overall. We also propose a new summarization model based on classifying articles by the leading sentence's strength.

Chapter 2 provides the background information necessary to understand summarization. We review prior work in designing summarization systems, explaining how their development lead to the current status of the field. Particular attention is given to more recent neural extractive models, as they serve as the basis for the following experiments.

Chapter 3 We explore BanditSum's reliance on positional cues through a series of perturbation experiments. After showing that positional cues play a dominating role in content selection, we present a method for countering the detrimental effects of lead bias. The method estimates the value of each sentence in an article, and modifies an existing model by encouraging it to match the estimated values. We show that this technique leads to summaries that are significantly more similar to reference summaries.

Chapter 4 We devise a new summarization model based on classifying articles by the strength of their leading sentences. We show that by training on a subset of articles with weak leading sentences, the resulting model outperforms a baseline trained on the full dataset on this subset. However, the classification step proves to be very difficult. Although the classifiers we build surpass a random baseline, they are not effective enough to achieve summarization performance gains.

Chapter 5 summarizes this work's main findings, and provides some potential directions for future research.

1.2 Statement of Contributions

This work was heavily influenced by colleagues' experiments, and for completeness, we include these motivating experiments here. In particular, the perturbation experiments in Chapter 3 – Section 3.2 – were conceived and executed by Yue Dong. Original contributions from this thesis include all other experiments in Chapter 3 and all experiments in Chapter 4, i.e. the auxiliary loss and the lead classifier sections.

Parts of this thesis also appeared in a paper published in the *2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing* (EMNLP-IJCNLP 2019). The paper, *Countering the Effects of Lead Bias in News Summarization via Multi-Stage Training and Auxiliary Losses* (Grenander et al. 2019), contains all experiments from Chapters 3.

2

Background

In this chapter, we provide an overview of select extractive summarization systems and other relevant concepts. Generally, an automatic summarization system is provided with a document D and returns a summary $\mathcal{S}$ which should satisfy certain properties, such as faithfulness to the original article, non-redundancy and coherence.

In abstractive summarization, the summary is generated token-by-token: the model typically maintains a vocabulary V, and at each time step t, it selects a word $w_t \in V$ as the next token. Abstractive summarization techniques typically must contend with large search spaces due to vocabulary sizes that are at least tens of thousands in size. Extractive summarization avoids this difficulty by creating summaries exclusively from text snippets in the source document. In the case where these snippets are sentences, an extractive summarizer can be viewed as assigning labels $y_i \in \{0, 1\}$ for each sentence s_i in D, indicating whether s_i will be included in the final summary or not.

Summarization tasks can also differ in the number of source documents provided. In single-document summarization, a single article is summarized, whereas in multi-document summarization, the system must condense multiple, possibly overlapping sources of information. Multi-document summarization adds an extra layer of difficulty due to the high-level of redundancy across documents. In this work, we focus on the single-document setting.

Although this thesis focuses on news summarization, many datasets exist for other summarization domains. Some notable examples include legal domains such as patents and

legislative documents (Sharma, Li, and Wang 2019; Kornilova and Eidelman 2019), medical documents (Kedzie, McKeown, and Daume III 2018; Zhang et al. 2018) and meeting notes (Carletta et al. 2005). Although these corpora provide exciting challenges, they are outside the scope of this work.

In this overview, we broadly cover (1) models, (2) datasets and (3) evaluation measures relevant for extractive summarization. We assume background knowledge on several basic machine learning and NLP concepts, including:

- Neural networks: backpropagation, gradient descent, multi-layer perceptrons (MLP), long short-term memory networks (LSTMs), convolutional neural networks (CNNs).
- Natural Language Processing: ngrams, stemming, tf–idf, word embedding techniques such as Word2Vec and GloVe (Mikolov et al. 2013; Pennington, Socher, and Manning 2014).

2.1 Models

Early summarization methods commonly use handcrafted features with manually tuned weights to score and rank salient sentences from the source article. Later, more sophisticated methods started borrowing methodologies from various other optimization algorithms such as PageRank and integer linear programming. As more labelled data became available, neural network-based methods such as Cheng and Lapata 2016 began to show prominence. In addition to better summarizing performance overall, neural methods are attractive as they typically require less feature engineering.

It is important to note that the methods we present here represent a small select set of representative algorithms, and is not reflective of the entire broad field of summarization. The most relevant group of summarization systems to our approach – neural network-based summarizers – is presented in Section 2.1.7.

The non-neural methods we highlight here are all unsupervised, except for the determinantal point process approach from Kulesza, Taskar, et al. 2012. Although non-neural, supervised summarization models do exist, many non-neural methods predate the creation of massive, high-quality, labelled summarization datasets. This lack of labelled data often put supervised methods at a disadvantage, and they do not feature as predominantly as unsupervised methods from this era. On the other hand, all neural summarization methods we discuss follow a supervised approach. In general, many neural network formulations – though not all – required labelled data, and are therefore usually supervised methods.

2.1.1 Early Methods

The earliest summarization system produces abstracts for scientific papers by computing a significance factor for each sentence (Luhn 1958). The algorithm first removes non-content words such as 'is' or 'and' using a lookup table. It then stems and sorts the remaining words by frequency, marking words that rank above a threshold as significant. The significance factor of a sentence is then computed as the ratio of significant words to sentence length. If the significance factor surpasses a certain threshold, it is included in the final summary.

Edmundson expanded on Luhn's work with a summarization system that incorporated 4 features in its decisions (Edmundson 1969). It considers presence of certain cue words such as 'significant' and 'impossible', word frequency of non-cue words, words from the article's title and heading, and word position. A weighted linear combination of the 4 features is computed for each sentence after manually determining suitable weights, and the top-ranked sentences are chosen as the summary.

2.1.2 Maximum Marginal Relevance

Maximum Marginal Relevance (MMR) is another well-known summarization algorithm, especially for its ability to create summaries in a multi-document setting (Carbonell and Goldstein 1998). It measures relevance and novelty independently, then greedily selects sentences using

a linear combination of these two metrics. The MMR formula is computed as:

$$MMR = \underset{D_i \in R \setminus S}{\operatorname{argmax}} \left[\lambda Sim_1(D_i, Q) - (1 - \lambda) \max_{D_j \in S} Sim_2(D_i, D_j) \right] \tag{2.1}$$

where R is a collection of documents, S is the summary so far, D_i is a sentence in $R \setminus S$, Q is a query vector, λ is a hyperparameter and Sim_1, Sim_2 are similarity metrics, possibly identical. The algorithm greedily maximizes this formula for each sentence in the source document until the target length is achieved.

2.1.3 MEAD

The MEAD summarization system also focuses on the multi-document setting, but with a document-clustering approach (Radev, Jing, and Budzikowska 2000). It relies on a document-clustering algorithm named CIDR to cluster same-topic documents together. CIDR produces a vector for each cluster identified, representing the words relevant for each particular cluster. The MEAD algorithm then computes 4 features for each sentence in the source documents and scores them using a linear combination with manually-tuned weights:

$$Score(S_i) = w_c C_i + w_p P_i + w_f F_i - w_R R_i \tag{2.2}$$

where C_i is a centroid value denoting the sentence's similarity to CIDR's clusters, P_i is a positional value ranking earlier-occurring sentences higher, F_i measures the degree of overlap with the document's first sentence and R_S is a redundancy penalty term similar to MMR. Similar to MMR, sentences that maximize this score are greedily chosen until a given compression ratio is achieved.

2.1.4 Graph-based Summarization

Numerous summarization approaches have benefitted from graph-based approaches (Mihalcea and Tarau 2004; Erkan and Radev 2004); we detail one of the most popular models, TextRank (Mihalcea and Tarau 2004). TextRank differs hugely from Edmundsonian methods, avoiding manually-tuned weights through innovative use of the PageRank algorithm.

It represents a document as a graph: sentences are represented as vertices, with weighted edges between sentences determined by lexical overlap. The iterative PageRank algorithm is then applied to this graph, outputting a relevance score for each sentence. The top-ranked sentences are selected to form the final summary.

2.1.5 ILP-based Summarization

Another family of summarization models is based around optimizing an integer linear programming problem (McDonald 2007; Clarke and Lapata 2008; Gillick and Favre 2009). We explain McDonald 2007's approach, one of the earliest ILP-based summarization models. He formulates summarization as a global inference problem based on maximizing relevance while minimizing redundancy, subject to a length constraint. Given a document $\mathbf{D} = \{t_1, \ldots, t_n\}$, a relevance function *Rel*, a redundancy function *Red*, and length constraint K, the optimal summary can be computed as:

$$S = \underset{S \subseteq \mathbf{D}}{\operatorname{argmax}} \sum_{t_i \in S} Rel(i) - \sum_{t_i, t_j \in S,\ i<j} Red(i,j) \tag{2.3}$$

$$\text{such that} \sum_{t_i \in S} l(i) \leq K \tag{2.4}$$

where $l(i)$ is the length of t_i. The formulation does not depend on specific *Rel* and *Red* functions, leaving the implementation up to the user. In their experiments, the authors use tf–idf vectors to represent sentences and measure redundancy and relevance with cosine similarity.

After demonstrating that the problem is NP-hard, the author then formulates 2 approximate solutions using a greedy approach similar to MMR and dynamic programming. An exact solution can also be extracted by using an integer linear programming approach, where sentence selection is formulated as a set of linear constraints.

2.1.6 Probabilistic Methods

SumBasic (Nenkova and Vanderwende 2005) is another well-known unsupervised summarization method. SumBasic assigns an initial probability to each word in a document based on word frequency, and ranks each sentence by the average word probability. After choosing the top-ranked sentence, SumBasic tackles redundancy issues by squaring the probabilities of each word appeared in the chosen sentence, thereby reducing its likelihood of appearing in subsequent sentence rankings:

$$p_{new}(w_i) = p_{old}(w_i) \cdot p_{old}(w_i) \text{ for all } w_i \in S_i \tag{2.5}$$

where S_i is the sentence chosen at time step i. This process is repeated until the desired summary length is achieved.

The last non-neural summarization method we discuss is based on determinantal point processes (DPP) (Kulesza, Taskar, et al. 2012). DPPs define a special type of probability measure over subsets of a fixed set of N elements. An essential characteristic of DPPs is that similar elements tend not to co-occur. In other words, DPPs promote diverse subsets by assigning low probability to similar pairs of elements. This characteristic lends itself naturally to extractive summarization, where non-redundancy is an important aspect. Kulesza, Taskar, et al. 2012 adapt DPPs to extractive summarization by computing feature vectors for each sentence in a document and training a DPP in a supervised setting. They show that DPPs are an effective method for extracting diverse, representative summaries from a document.

2.1.7 Neural Summarization

In recent years, many researchers have shifted towards summarization methods based on deep learning, where lexical representations are learned by neural networks. Although in general these models may require more computational resources and large amounts of data to train, neural models are often considered more effective summarizers provided that the evaluation domain is similar to the training setting (Nallapati, Zhai, and B. Zhou 2017).

Neural-based extractive summarization methods typically comprise two steps. In the **sentence representation** step, models map raw text into some abstract representation, usually a vector. Then, in **sentence selection**, models use the content representations to rank and select which sentences should constitute the summary.

Kågebäck et al. 2014 presented one of the earliest extractive neural summarization models, called **Continuous Vector Space Models**. The model first uses CW or Word2Vec vectors to represent words in the document (Mikolov et al. 2013; Collobert and Weston 2008). Sentence representations are then created by summing a sentence's word embeddings or using a recursive auto-encoder (RAE) to combine word embeddings. The RAE aims to compress word embeddings recursively until single vector is left, representing the whole sentence. Sentence selection is performed by following the Lin-Bilmes method, in which a linear combination of coverage and diversity factors is approximately optimized (H. Lin and Bilmes 2011).

Cheng and Lapata 2016 present a summarizer more tightly integrated with neural networks. After encoding the document's words with Word2Vec embeddings, they use a convolutional neural network (CNN) to create hidden representations $(s_1, \ldots, s_m)$. In order to capture latent temporal information, these vectors are subsequently fed into a long short-term memory (LSTM) network to achieve sentence embeddings $(h_1, \ldots, h_n)$. To extract sentences, another LSTM followed by a multi-layer neural network is used to predict which sentences should form the summary. The extractor sequentially labels each sentence, using the previous time step's decision to help inform whether or not to include the current sentence in the final summary:

$$\bar{h}_t = \text{LSTM}(p_{t-1}s_{t-1}, \bar{h}_{t-1}) \tag{2.6}$$

$$p(y_t = 1|D) = \sigma(\text{MLP}([\bar{h}_t; h_t])) \tag{2.7}$$

where $[\bar{h}_t; h_t]$ denotes concatenation of the two vectors.

Large neural models ordinarily require large amounts of training data, which was not readily available at the time. To overcome this data paucity, the authors retrieved hundreds

of thousands of news articles from the Daily Mail news archives, along with associated bullet point highlights to serve as summaries. The model is then trained using this resource.

Nallapati, Zhai, and B. Zhou 2017's **SummaRuNNer** model employs two layers of bidirectional GRUs to create sentence representations. The first biGRU runs over the document's words to create word-level representations. Each sentence's word-level representations are then averaged and fed into another GRU to create sentence-level representations $(h_1, \ldots, h_n)$. A document representation $\mathbf{d}$ is also computed by averaging sentence embeddings followed by a non-linear transformation.

A logistic layer then classifies whether to include each sentence or not based on a variety of factors:

$$p(y_j = 1|h_j, s_j, \mathbf{d}) = \sigma\left(W_c h_j + h_j^T W_s \mathbf{d} - h_j^T W_r \tanh\left(s_j\right) + W_{ap} p_j^a + W_{rp} p_j^r + b\right) \tag{2.8}$$

where the W_x and b variables represent learnable parameters. The "summary-so-far" representation s_j is recursively computed by weighting each sentence representation by the probability outputted by the logistic classifier:

$$s_j = \sum_{i=1}^{j-1} h_i p(y_i = 1|h_i, s_i, d) \tag{2.9}$$

Finally, positional embeddings p_j^a and p_j^r indicating the absolute and relative sentence position are included as inputs to the logistic classifier.

Narayan, Cohen, and Lapata 2018b argue that minimizing a cross-entropy loss objective, as done in many previous neural models, is not ideal for supervised summarization tasks. In order to use cross-entropy loss for supervised summarization tasks, it is often necessary to create heuristic gold labels. Narayan, Cohen, and Lapata 2018b note that mismatches exist between heuristic labels and true summary-worthy content, which may hurt summarization performance. Instead, they argue that directly optimizing ROUGE can prevent these errors. In order to optimize ROUGE, the authors formulate a novel objective based on the REINFORCE algorithm (Williams 1992). Their model architecture itself is similar to Cheng and Lapata 2016: words are first encoded by CNN followed by a LSTM to create sentence

embeddings. An LSTM followed by a softmax layer then assigns a probability score to each sentence. During training, sentences with low ROUGE scores are manually filtered to reduce the search space, and a summary is created by sampling from the remaining probabilities. After creating a summary hypothesis, the model is scored against the reference summary using ROUGE, and this score is backpropagated through the network using REINFORCE.

BanditSum is another recent neural model trained using a reinforcement learning-based objective function (Dong et al. 2018). Unlike Narayan, Cohen, and Lapata 2018b, BanditSum does not filter the action space, and thereby samples from the true action space instead of approximating it. BanditSum employs two sets of bidirectional LSTMs to create sentence representations. After embedding words with GloVe (Pennington, Socher, and Manning 2014), a word-level LSTM creates word representations of the article. For each sentence, the word-level representations are averaged and inputted into a sentence-level LSTM to create sentence representations $(h_1, \ldots, h_n)$. A multi-layer perceptron then runs over the sentence representations to create *sentence affinity* scores for each sentence. Similar to Narayan, Cohen, and Lapata 2018b, hypothesis summaries are created by sampling without replacement from the sentence affinities. After sampling B distinct summaries, the extracts are scored using ROUGE and the weights are updated using the REINFORCE algorithm (Williams 1992).

2.1.7.1 Language Model Pre-Training

Recently, summarization methods based on language model pre-training objectives have achieved state-of-the-art results. These approaches, released concurrently with our work, are particularly interesting because they tend to exhibit less lead bias, and thus provide promising directions for future research. We detail here the extractive model BertSum (Liu and Lapata 2019), though abstractive approaches based on large pre-training objectives also exist (Lewis et al. 2019).

BertSum is an extractive model that uses BERT to encode sentences (Liu and Lapata 2019). BERT is a language representation model built with a Transformer architecture

(Vaswani et al. 2017) and trained with a masked language modeling task. It has proven effective as an encoder for both words and sentences in many NLP tasks. Liu and Lapata 2019 use the pre-trained BERT model to encode sentences in a document, followed by a second Transformer to extract and rank sentences by summary-worthiness. The model is fine-tuned with a cross-entropy loss using heuristically-generated gold labels.

2.2 Summarization Evaluation

2.2.1 Human Evaluation

Traditionally, evaluating summary quality has been performed by human judges. Reference and systems summaries are compared along multiple dimensions, such as coverage, non-redundancy and coherence. However, although human summary evaluation is usually considered to be the most conclusive method to compare summarization systems, it is often difficult to scale human summary evaluation to tens of thousands of summary judgements. Moreover, human judgement has not been standardized in the summarization community, and different authors often use incompatible metrics when evaluating summary quality. These inconsistencies complicate comparisons between different systems.

2.2.2 ROUGE

Driven by the difficulties of human evaluation, automatic summary evaluation has been researched for many years. The most common evaluation scheme is **ROUGE**, or **Recall-Oriented Understudy for Gisting Evaluation** (C.-Y. Lin 2004), a set of metrics based on measuring word overlap between the generated system summary and a reference summary. ROUGE is comprised of several individual measures, representing distinct aspects of comparison. In our experiments, we report the following metrics:

- ROUGE-N measures the ngram overlap between the generated and reference summaries. Given a set of reference summaries $\mathcal{R} = \{\mathcal{R}_1, \ldots, \mathcal{R}_m\}$ and a system summary

$\mathcal{S}$, the ROUGE-N score is computed as:

$$\text{ROUGE-N}(\mathcal{S}, \mathcal{R}) = \frac{\sum_{i=1}^{m} \sum_{gram_N \in \mathcal{R}_i} Count_{match(\mathcal{S},\mathcal{R}_i)}(gram_N)}{\sum_{i=1}^{m} \sum_{gram_N \in \mathcal{R}_i} Count(gram_N)} \tag{2.10}$$

where N is the ngram length we are considering, $gram_N$ is an ngram of length N, and $Count_{match(\mathcal{S},\mathcal{R}_i)}$ is the number of matching ngrams between $\mathcal{R}_i$ and $\mathcal{S}$.

Although the original paper describes this metric as a recall-based measure, recent models have at times reported the F_1 score instead. This decision is usually based on the dataset and whether a summary length limit is specified. To better compare with state-of-the-art models, we compute the F1 score. In our experiments, we report both ROUGE-1 (unigram) and ROUGE-2 (bigram) metrics.

- ROUGE-L scores how well the word order is respected in the reference summary. It is based on a "*union* longest common subsequence" score $LCS_\cup$ between the system summary and each sentence in the reference summary (C.-Y. Lin 2004). Given a system summary $\mathcal{S}$ with sentences $s_1, \ldots, s_v$ and a reference summary $\mathcal{R}$ with sentences $r_1, \ldots, r_u$, we first create sets from the longest common subsequence (LCS) between each reference sentence r_i and $\mathcal{S}$. $LCS_\cup(r_i, \mathcal{S})$ is computed by taking the union of these sets and dividing by the total word length of r_i:

$$LCS_\cup(r_i, \mathcal{S}) = \frac{|\bigcup_{i=1}^{u} LCS(r_i, \mathcal{S})|}{|r_i|} \tag{2.11}$$

The recall and precision scores are then computed as:

$$R_{lcs} = \frac{\sum_{i=1}^{u} LCS_\cup(r_i, \mathcal{S})}{|\mathcal{R}|} \quad , \quad P_{lcs} = \frac{\sum_{i=1}^{u} LCS_\cup(r_i, \mathcal{S})}{|\mathcal{S}|} \tag{2.12}$$

The final ROUGE-L F1 score is given by the F1 score of the above precision and recall.

2.3 Datasets

The Document Understanding Conferences[1] (DUC) produced an annual summarization dataset from 2001–2007, before joining as a track within the Text Analysis Conferences[2] (TAC) (Har-

[1] https://duc.nist.gov/
[2] https://tac.nist.gov/

man and Over 2004; Dang 2006; Dang and Owczarzak 2008). The DUC/TAC datasets consist of news articles paired with multiple human-written summaries, making these datasets an excellent resource for multi-document summarization. However, the datasets generally do not contain enough samples to train neural network-based summarization models.

The CNN / Daily Mail dataset is a large collection of news stories with associated bullet point highlights used as summaries (Hermann et al. 2015). The dataset is split into 287,227/13,368/11,490 article-summary pairs for the training / development / testing datasets, making it a tremendous resource for neural methods which often require large quantities of training data for adequate generalization. The CNN / Daily Mail is especially suitable for extractive systems as the summaries are known to be quite extractive in nature – in other words, summary content is often copied from the source document (Grusky, Naaman, and Artzi 2018).

While other news summarization datasets exist, such as the New York Times Corpus, Newsroom and XSum (Sandhaus 2008; Grusky, Naaman, and Artzi 2018; Narayan, Cohen, and Lapata 2018a), they are more abstractive and hence less suitable for our purposes. Gigaword is another well-known summarization dataset containing millions of news articles from several publishers (Napoles, Gormley, and Van Durme 2012); however, the news articles are not paired with summaries, complicating the use of machine learning approaches. In all experiments, we report results using the CNN / Daily Mail dataset.

3

Countering Lead Bias via Auxiliary Loss

A recent study by Kedzie, McKeown, and Daume III 2018 shows that MLE-based summarization models learn a significant bias towards selecting early sentences when trained on news articles, a phenomenon they term 'lead bias'. They demonstrate that in some models, as much as 58% of selected summary sentences come directly from the article's leading sentences. Moreover, when these models are trained on news articles whose sentences are randomly shuffled, the performance drops considerably on the news domain.

These results suggest that lead bias is a significant bottleneck for news summarization systems. For this reason, we would like to explore the severity of this issue further and design potential solutions. In this chapter, we corroborate and expand on Kedzie, McKeown, and Daume III 2018's findings by perturbing sentence order in news articles in a multitude of ways. These experiments confirm that positional cues dominate the learning signals for state-of-the-art summarization systems.

We then detail our method for countering lead bias, based around augmenting an existing summarization system with an auxiliary loss objective aimed at properly evaluating sentence relevance independent of sentence position. We demonstrate that this method is able to decrease the percentage of time the base model selects leading sentences, while also improving the model's overall summary quality. The resulting model is particularly more adept at summarizing articles where the lead performance is weak, indicating that our method is effective at helping models look beyond simple positional cues.

3.1 Base model: BanditSum

In order to experiment with lead bias in modern summarization models, we first need to choose a model to analyze. Here, we detail BanditSum's formulation (Dong et al. 2018), and why it lends itself well to lead bias experiments.

BanditSum approaches extractive summarization as a contextual bandit problem. This approach views the document as a context, while selecting a subset of sentence indices corresponds to an action. Sentences are first mapped to *sentence affinity* scores, representing the model's propensity to include each sentence in its summary. BanditSum then takes advantage of reinforcement learning techniques, interpreting these affinity scores as a stochastic policy. The model uses the REINFORCE algorithm (Williams 1992) to compute policy gradient updates, avoiding the MLE-based summarization issues raised by Narayan, Cohen, and Lapata 2018b. The authors also confirm empirically that BanditSum's non-autoregressive framework enables it to sidestep some lead bias issues faced by other autoregressive summarization systems.

3.1.1 Model Architecture

Given a document D with n sentences, the neural model produces sentence affinity scores $\pi_\theta = \{\pi_1, \ldots, \pi_n\}$, where $\pi_i \in [0, 1]$. After first using GloVe (Pennington, Socher, and Manning 2014) to embed each word, a word-level LSTM captures interword dependencies. In each sentence, these word representations are averaged and then passed to a sentence-level LSTM to create sentence features $h_1, \ldots, h_n$. Finally, a multi-layer perceptron decoder maps $h_1, \ldots, h_n$ to the sentence affinity scores $\pi_\theta = \{\pi_1, \ldots, \pi_n\}$.

Using the sentence affinity scores, BanditSum now computes a gradient update following the REINFORCE algorithm (Williams 1992). In particular, hypothesis summaries are repeatedly sampled according to π_θ, consisting of K sentences each.[1] Each hypothesis summary

[1]The authors use $K = 3$ following the average reference summary sentence length.

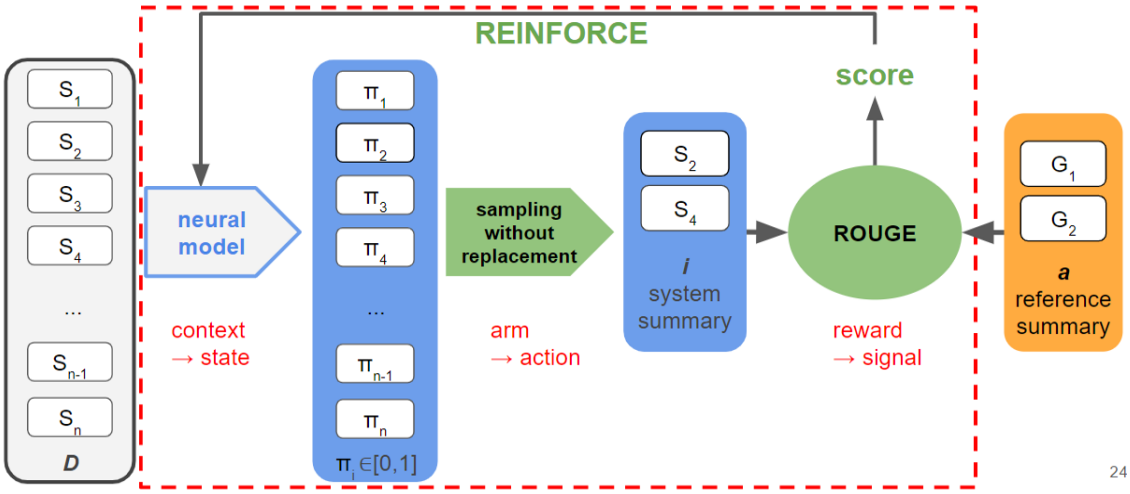

Figure 3.1: An overview of the BanditSum model. Sentence affinity scores are sampled to create hypothesis summaries, which are then scored against a reference summary using ROUGE. A gradient update is then computed using REINFORCE. Figure created by Yue Dong and used with permission.

is then scored against the reference summary using a reward function based on ROUGE:

$$\mathrm{R}(\mathcal{S}, \mathcal{R}_D) = \frac{1}{3}\left(\text{ROUGE-1}_{\mathrm{F1}}(\mathcal{S}, \mathcal{R}_D) + \text{ROUGE-2}_{\mathrm{F1}}(\mathcal{S}, \mathcal{R}_D) + \text{ROUGE-L}_{\mathrm{F1}}(\mathcal{S}, \mathcal{R}_D)\right) \quad (3.1)$$

where $\mathcal{S}$ is the system summary and $\mathcal{R}_D$ is the reference summary. The reward function scores relevance and redundancy simultaneously by using an F1 version of ROUGE. Finally, using a self-critical baseline $\overline{r}$ for variance reduction (Rennie et al. 2017), the policy gradient update is computed as follows:

$$\nabla_\theta J(\theta) = \frac{1}{B}\sum_{i=1}^{B} \nabla_\theta \log p_\theta(\mathcal{S}_i|D)\left[\mathrm{R}(\mathcal{S}_i, \mathcal{R}_D) - \overline{r}\right] \quad (3.2)$$

where $p_\theta(\mathcal{S}_i|D)$ is the joint probability of including the sentences in $\mathcal{S}_i$ (i.e. according to π_θ) and B is the batch size. This equation corresponds to the update equation given by REINFORCE (Williams 1992).

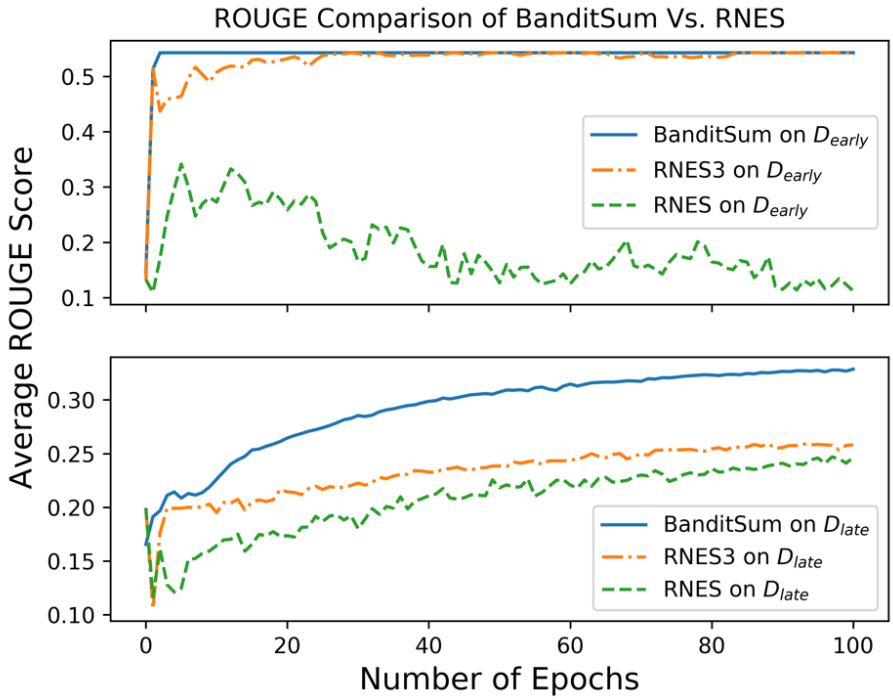

Figure 3.2: Comparison of BanditSum and RNES on D_{early} and D_{late} subsets. While performance is roughly similar on D_{early}, BanditSum dominates over RNES on D_{late}. Figure from Dong et al. 2018.

3.1.2 Analysis

One of BanditSum's key properties is that the model is **non-autoregressive**, meaning that the decision to include sentence s_i does not explicitly depend on sentences $s_1, \ldots, s_{i-1}$. An example of an autoregressive model is given by SummaRuNNer (Nallapati, Zhai, and B. Zhou 2017), whose "summary-so-far" representation recursively depends on previous summary decisions (see Equations 2.8 and 2.9).

BanditSum's authors conjecture that autoregressive models are more likely to suffer from lead bias issues. Autoregressive models must decide whether to include early sentences before fully evaluating later sentences, and this may lead the models to erroneously include earlier sentences in their summaries. To verify this claim, the authors compare BanditSum to an autoregressive model RNES (Wu and Hu 2018), which also uses a reinforcement learning-based, neural system. The authors annotate articles from the validation set by an extractive

index $\overline{idx}$, denoting whether an oracle-generated extractive summary appears earlier or later in the document. Articles are then sorted according to $\overline{idx}$, and two datasets are created. D_{early} is formed from the top 50 documents w.r.t. $\overline{idx}$ (i.e. best summary appears earlier), while D_{late} contains the lowest-scoring 50 articles (i.e. best summary appears late). Finally, both BanditSum and RNES are trained on the D_{early} subset then evaluated on the D_{late} subset. The results are displayed in Figure 3.2.

While BanditSum and RNES perform similarly on D_{early}, BanditSum eclipses RNES on D_{late}, converging much quicker to higher scoring summaries. Owing to its non-autoregressive formulation, BanditSum can produce more effective summaries when summary-worthy sentences occur later in the article. This fact provides a good basis for performing lead bias experiments with BanditSum, as we would prefer a model that is not inherently biased towards earlier occurring sentences.

3.2 Lead Bias of News Systems

While the observed performance drops in Kedzie, McKeown, and Daume III 2018 may be due to the destruction of position cues, they may also arise because the article's coherence and context were lost, i.e. shuffling sentence order affects semantics. We explore this phenomenon more comprehensively by distorting sentence order in a fine-grained approach.

We manipulate the CNN / Daily Mail dataset to preserve sentence position information at different levels. For each setting, we train separate instances of BanditSum, then test the model on the other datasets. In the **random** setting, sentences are shuffled randomly; in **reverse**, they are in reverse order; in **insert-lead** and **insert-lead3**, we insert an out-of-document sentence (chosen randomly from the corpus) as the first sentence or randomly as one of the first three sentences, respectively. Finally, **original** preserves the original ordering.

In Table 3.1 and 3.2, we show BanditSum's performance when trained and tested on the various datasets. All models (except random) perform worse when tested on a mismatched data perturbation. Although some drops in performance are expected, the extreme ROUGE

Train setting \ Test setting	original	random	reverse	insert-lead	insert-lead3
Lead-3 baseline	32.68	22.81	17.94	27.67	27.68
original	**33.85**	26.18	20.71	31.71	31.11
random	30.88	**29.70**	29.79	29.97	30.09
reverse	21.35	26.32	**33.59**	21.63	21.65
insert-lead	33.21	26.07	20.70	**33.41**	31.59
insert-lead3	32.29	25.57	20.22	32.92	**32.15**

Table 3.1: BanditSum's performance—calculated as the average between ROUGE-1,-2, and -L F1—on the CNN/Daily Mail validation set. The sentence position information is perturbed at different levels.

Train setting	Average of ROUGE-1, -2, -L	Standard Deviation
Lead-3 baseline	25.76	5.00
original	28.71	4.72
random	**30.09**	**0.42**
reverse	24.91	4.72
insert-lead	29.00	4.93
insert-lead3	28.63	4.98

Table 3.2: Average ROUGE-1, -2 and -L scores and standard deviation on the CNN / Daily Mail validation set using different training settings.

difference highlights how much more attention is given to position cues instead of semantic content. Even when the distortion is at a single lead position in **insert-lead** and **insert-lead3**, the model trained on the original setting suffers large performance drops. This drop occurs despite little semantic difference between the original and insert-lead/insert-lead3 settings. This experiment worryingly suggests that very slight changes to sentence position can have drastic effects on summarization models.

These results corroborate Kedzie, McKeown, and Daume III 2018's findings for RL-based systems. Similar results hold two other models we test: RNES (Wu and Hu 2018) drops 4.2 and Refresh (Narayan, Cohen, and Lapata 2018b) drops 3.4 points in average ROUGE when trained on shuffled data and tested on the original dataset.

The results in Table 3.1 are concerning as the large drops between mismatched train and test settings suggest that position cues are a dominating signal for BanditSum. These findings suggest that BanditSum may largely ignore semantic content in favour of cheap

positional cues. Obviously, solely exploiting positional cues will not work for news articles where summary-worthy content appears later, such as in Table 1.2. Beyond the news domain, other summarization domains may contain their own positional biases and similarly influence automatic learners. More generally, future summarization systems which attempt to create summaries across many domains cannot solely rely on positional cues if these signals change across domains. In order to be effective, systems must learn to balance between exploiting positional cues in the data and understanding the underlying semantic content.

Interestingly, the **random** model has the best mean performance and the lowest variation, indicating that removing or lessening position bias may allow a model to focus on learning robust sentence semantics. Following this observation, we explore novel techniques aimed at reducing the influence of lead bias on the learning process.

3.3 Auxiliary Loss Objective

Motivated by the experiments in Section 3.2, we set towards designing methods to reduce models' dependence on positional cues. We observe that in general, BanditSum tends to converge to a low-entropy policy, in the sense that the model's affinity scores are either 1 or 0 at the end of training. Regularizing low-entropy policies can increase a model's propensity to explore potentially good states or stay close to a known good policy (Nachum, Norouzi, and Schuurmans 2017; Galashov et al. 2019). We extend this idea to summarization by introducing a ROUGE-based loss which regularizes the model policy using an estimate of the value of individual sentences.

As the goal is to guide the model towards properly valuing sentences, we must first approximate the true value of each sentence in a document. These sentence-level estimates are computed as a categorical distribution P_R over the document:

$$P_R(x = i) = \frac{r(s_i, \mathcal{G})}{\sum_{j=1}^{n} r(s_j, \mathcal{G})} \tag{3.3}$$

where r is the average of ROUGE-1, -2 and -L F_1 scores between sentence s_i in the article and

the reference summary $\mathcal{G}$. Since ROUGE measures lexical overlap, this distribution provides a good approximation of each sentence's relevance to the reference answer.

We would like the model's predictive distribution $P_{\mathcal{M}}$ to approximately match P_R. To compute $P_{\mathcal{M}}$, we normalize the model's predicted sentence scores. In other words, if the model outputs sentence affinity scores $\pi_\theta = (\pi_1, \ldots, \pi_n)$, $P_{\mathcal{M}}$ is computed as:

$$P_{\mathcal{M}}(x = i) = \frac{\pi_i}{\sum_{j=1}^{n} \pi_j} \tag{3.4}$$

Our auxiliary loss is defined as the KL divergence: $\mathcal{L}_{\text{KL}} = D_{\text{KL}}(P_R \parallel P_{\mathcal{M}})$. We modify the update rule using a weighted sum of the original model loss and our KL-based loss:

$$\theta^{(t+1)} = \theta^{(t)} + \alpha \left(\nabla \mathcal{L}_{\mathcal{M}}(\theta^{(t)}) + \beta \nabla \mathcal{L}_{\text{KL}}(\theta^{(t)}) \right) \tag{3.5}$$

Here, $\theta^{(t)}$ represents the model's parameters at time step t, $\mathcal{L}_{\mathcal{M}}$ is the original model's loss function, α is the learning rate, and β is a hyperparameter. In the case of BanditSum, $\mathcal{L}_{\mathcal{M}}$ corresponds to $J(\theta)$ from Equation 3.2.

Notably, $\mathcal{L}_{\text{KL}}$ does not account for redundancy, and we do not use this measure as the sole objective function. In fact, experiments we conduct show that solely using the KL objective function does not result in improved performance.

3.4 Experimental Setup

We test our method using the CNN/Daily Mail dataset (Hermann et al. 2015), building on top of the author-provided BanditSum implementation. To reduce training time, we pre-compute and store the ROUGE-1, -2, and -L average for every sentence triplet of each article, using PyTables and HDF5 (PyTables Developers Team 2002-2019; The HDF Group 1997-2019). This allows for a considerable increase in training speed. We limit the maximum number of sentences considered in an article to the first 100. All the models were trained for 4 epochs. We set the auxiliary loss hyperparameters $\alpha = 1e-4$ and $\beta = 0.0095$ in Equation 3.5 based on a grid search using the Tune library (Liaw et al. 2018).

Model	ROUGE			Lead Overlap
	1	2	L	%
Lead-3	40.06	17.53	36.18	100.0
Oracle	56.53	32.65	53.12	27.24
Refresh	40.0	18.2	36.6	–
NeuSum	40.15	17.80	36.63	58.24
RNES	41.15	18.81	37.75	68.44
BanditSum	41.68	18.78	38.00	69.87
B.Sum+pretrain	41.68	18.79	37.99	70.77
B.Sum+entropy	41.71	18.87	38.04	64.83
BanditSum+KL	**41.81***	**18.96***	**38.16***	65.13

Table 3.3: ROUGE scores for systems. Lead overlap denotes the model's overlap in extraction choices with the lead-3 baseline. Scores significantly higher than BanditSum with $p < 0.001$ (bootstrap resampling test) are marked with *. Note that we are unable to evaluate Refresh on the lead overlap measure due to lack of access to the model outputs.

We also train a baseline entropy model by replacing $\mathcal{L}_{\text{KL}}$ with the negated entropy of $P_{\mathcal{M}}$ in Equation 3.5. This loss penalizes low entropy, helping the model explore without indicating which sentences are relevant. In this sense, it is an 'undirected' loss function compared to our proposed method. In contrast, the $\mathcal{L}_{\text{KL}}$ loss provides an indication of each sentence's value, and provides a more targeted signal. We present the results of Lead-3 baseline (first 3 sentences), and two other competitive models – Refresh and NeuSum (Narayan, Cohen, and Lapata 2018b; Q. Zhou et al. 2018). Lastly, we include results from an oracle summarizer, computed as the triplet of source sentences with the highest average of ROUGE-1, -2 and -L scores against the abstractive gold standard.

3.5 Results

Table 3.3 reports the F1 scores for ROUGE-1,-2 and -L (C.-Y. Lin 2004). We use the pyrouge[2] wrapper library to evaluate the final models, while training with a faster Python-only implementation[3]. We test for significance between the baseline models and our proposed techniques using the bootstrap method. This method was first recommended for testing

[2] www.github.com/bheinzerling/pyrouge
[3] www.github.com/Diego999/py-rouge

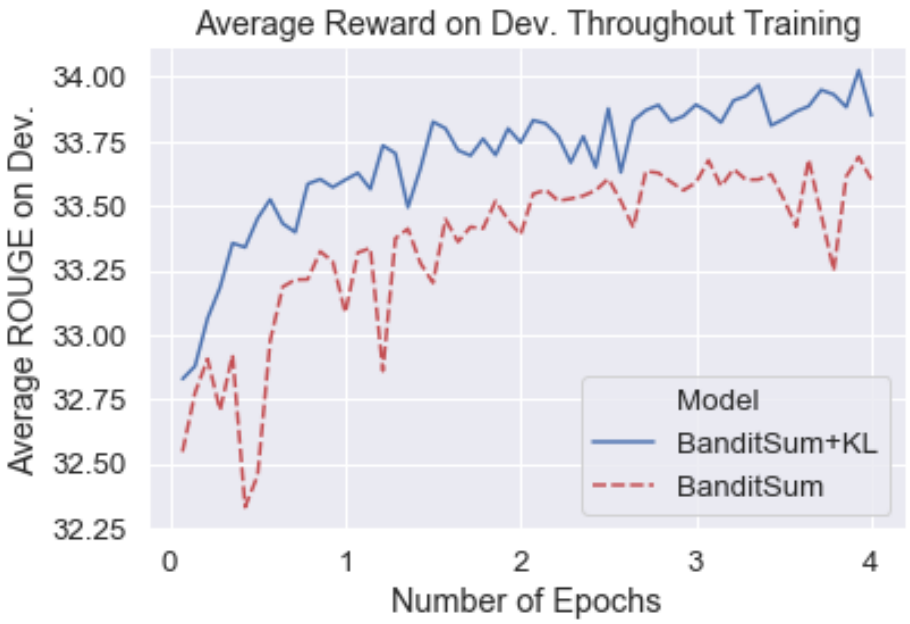

Figure 3.3: Training curves for BanditSum-based models. Average ROUGE is the average of ROUGE-1, -2 and -L F1.

significance in ROUGE scores in the original ROUGE paper (C.-Y. Lin 2004), and has subsequently been advocated as an appropriate measure in works such as Dror et al. 2018 and Berg-Kirkpatrick, Burkett, and Klein 2012.

The simple entropy regularizer has a small but insignificant improvement, indicating that while boosting exploration is helpful, it is too simple to provide real performance gains. In contrast, the BanditSum+KL method significantly improves over BanditSum, with an extra 0.15 ROUGE points on average. The last column reports the percentage of summary sentences which overlap with the lead. The auxiliary loss leads to a 4.7% absolute decrease in such selections compared to the base system, while also reaching a better ROUGE score. Figure 3.3 shows that the reward for the auxiliary loss model is consistently above the base.

3.6 Analysis

Given that our method is designed to alleviate overfitting to positional cues, we suspect that the BanditSum+KL model achieves greater gains when the lead constitutes a poor summary.

To test this idea, we examine the auxiliary loss model on documents where the summary is mostly comprised of lead sentences D_{early}, mostly sentences much later in the article D_{late}, and a dataset at the midway point, D_{med}. To create these sets, we compute an extractive index $\overline{idx}$ for each test set document, similar to Dong et al. 2018. Given a document, we use the oracle summarizer's extracted indices (i, j, k) and compute $\overline{idx}$ as:

$$\overline{idx} = \frac{i + j + k}{3n} \tag{3.6}$$

where n is the number of sentences in the document.

After the test articles are ranked using the $\overline{idx}$ metric, the 100 test articles with lowest average index are D_{early}, the 100 with highest value are D_{late} and the 100 closest to the median are D_{med}. In Table 3.4, we can see that the auxiliary loss model's improvements are even more amplified on D_{med} and D_{late}.

The second line in Table 3.4 reports the oracle ROUGE scores of the best possible extractive summary. While all systems are quite close to the oracle on D_{early} they only reach half the performance on D_{late}. This gap indicates that our improvements only scratch the surface, but also that this problem is worthy and challenging to explore.

In Figure 3.4, we compare BanditSum and BanditSum+KL's average affinity scores. The BanditSum+KL method sharply reduces the average affinity score on the first two sentence positions, while increasing the average affinity for sentence positions 3 and beyond. This indicates that our method can help the model reach sentences further down the document. We notice similar trends for the average position selection, though the BanditSum+KL method remains far from the oracle summarizer.

We also compare model prediction distributions on an example article from the validation set in Figure 3.5. While the BanditSum predictions forms a very low-entropy distribution, the KL loss helps even out the predictions and reach sentences further in the document.

It is worth noting that we have attempted to build a single model which can summarize both lead-biased articles and those whose information is spread throughout. Our aim was to encourage the model to explore useful regions as a way of learning better document semantics.

Model	D_{early}	D_{med}	D_{late}
Lead-3	46.17	30.90	20.18
Oracle	50.52	47.92	42.21
NeuSum	40.70	31.26	20.44
RNES	41.76	32.11	20.62
BanditSum	43.10	32.65	21.63
BanditSum+entropy	41.96	32.59	22.12
BanditSum+KL	42.63	33.05	21.96

Table 3.4: Average ROUGE-1, -2 and -L F1 scores on D_{early}, D_{med} and D_{late} subsets. Each set contains 100 documents.

But we hypothesize that our models can be further improved by learning to automatically predict when the lead paragraph suffices as a summary, and when the model should look further in the document.

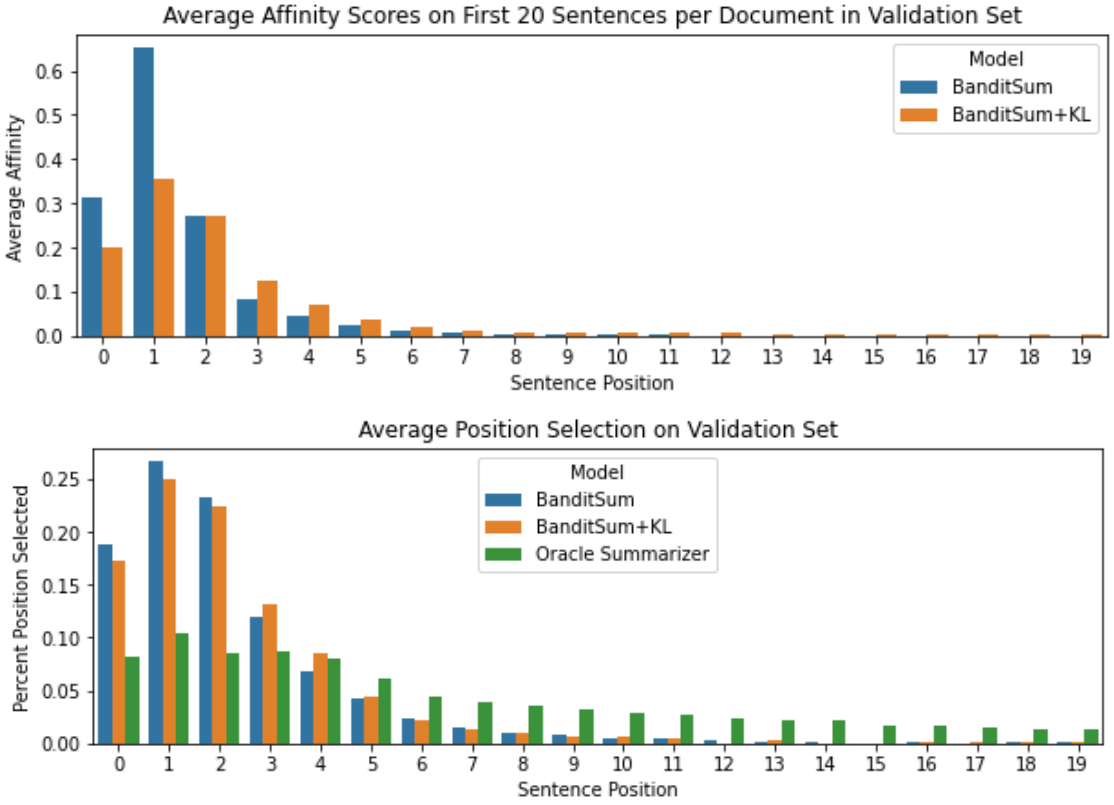

Figure 3.4: (Top) Average affinity scores for the BanditSum and BanditSum+KL models. Note that the original model has far higher affinity on average for the first two sentence positions, while our proposed model has a more even distribution. (Bottom) Average position selected for the same models. We observe similar trends as the affinity score distribution, though the BanditSum+KL method remains far from the oracle positions.

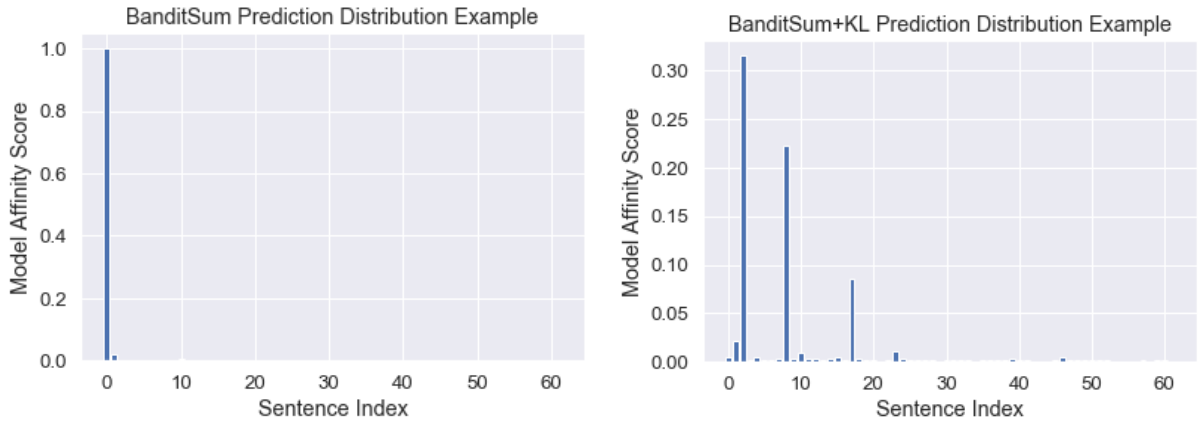

Figure 3.5: Example of the BanditSum prediction distribution (left) vs. the BanditSum+KL prediction distribution (right) on the same given article from the validation set. The original formulation is more prone to lead bias compared to our proposed method, as demonstrated here.

4

Summarizing by Classifying Lead Performance

Chapter 4's results suggest that methods addressing model dependence on positional cues can be an effective way to boost summarization performance. The previous chapter's auxiliary loss method is able to decrease how frequently the base model selects from leading sentences, while also significantly improving summary quality. Although the auxiliary loss method's summarization improvement is statistically significant, it does not offer extraordinary gains over baselines. Figure 3.4 reveals that the method only offers modest reductions in how often lead sentences are selected and does not approach the percentage an oracle summarizer selects from the lead.

The results do not necessarily suggest that addressing positional biases is an ineffective strategy. Previous works such as Kedzie, McKeown, and Daume III 2018 and our own experiments in Chapter 3 have already shown that positional cues represent a major bottleneck for learning summarization. Instead, we suspect that the auxiliary loss methods are not sufficiently aggressive enough to counter the effects of lead bias. For these reasons, we would prefer a model formulation that integrates the ideas of lead bias more centrally.

We hypothesize that articles with a strong lead (i.e. the lead forms an accurate summary) are fundamentally different from cases with a weak lead, such that a classifier could distinguish between these two cases. We look to build such a classifier, so that we can apply different summarization strategies for the two scenarios. In this chapter, we detail this new classification task, the model formulation, the challenges surrounding the task and the

resulting model's summarization capabilities.

4.1 Classification Task

We first define how to categorize articles based on the lead baseline's strength. Since there is a large variance between the highest achievable ROUGE score between different articles, we avoid categorizing articles solely based on the lead ROUGE score. For example, if the lead baseline achieves 30.0 ROUGE-1 on both articles A and B, but the oracle summarizer respectively achieves 60.0 and 30.0 ROUGE-1, we should avoid placing A and B in the same category. To avoid this scenario, we rank articles by how well the lead performs in relation to an oracle summarizer:

$$\text{Proportion Score}(D) = \frac{r_{lead}(D)}{r_{oracle}(D)} \tag{4.1}$$

where r_{lead} denotes the lead-3's ROUGE-1, -2 and -L average score on D, and r_{oracle} denotes the same quantity using an oracle summarizer. As previously, the lead is defined as the first 3 sentences in the document, while the oracle is defined in the same way as in Chapter 3.

We also need to define a threshold value T in order to divide articles. To determine an acceptable threshold value, we experiment with varying threshold values on the CNN / Daily Mail development set. For each threshold value, we divide the articles based on whether their proportion score falls below or above the given threshold. Documents below the threshold are summarized by the BandiSum+KL model, while documents above the threshold are handled by the lead-3 baseline. The results are shown in Figure 4.1.

The highest gains in ROUGE performance are attained when the threshold $T = 0.75$, and we therefore use this value as the threshold in the following experiments.

Since distinguishing points very close to the threshold may be difficult for classifiers, we also experiment with removing articles close to the threshold T. After ranking all articles by their proportion score, we create subsets by removing articles close to the threshold. Specifically, we experiment with removing 20, 30, 50 and 60 percent from each of the training

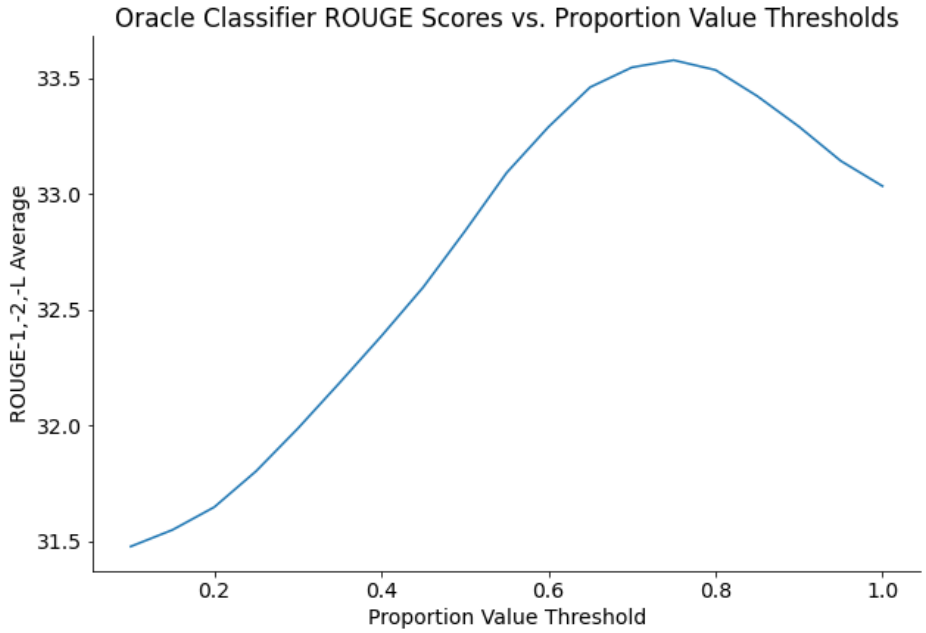

Figure 4.1: Each data point in this figure represents a trial with varying threshold T. In each trial, articles with proportion scores $< T$ are summarized by BanditSum+KL and otherwise summarized by the lead-3 baseline. The highest $\frac{1}{3}(R1 + R2 + RL)$ value is attained when $T = 0.75$.

/ development / testing data, centered around T. The subsets are named D_{20}, D_{30}, D_{50} and D_{60}. We use D_{all} to refer to the full dataset with no data removed. Furthermore, we designate the subset of articles with a proportion score less than T as D_{late}, and greater than T as D_{early}.

4.2 Models

In this section, we introduce the various components that comprise our summarization model. We describe (1) a strong baseline using BanditSum and BERT, (2) the neural classifier model, and (3) the BanditSum + $\text{BERT}_{\text{early}}$ and BanditSum + $\text{BERT}_{\text{late}}$ models for the D_{early} and D_{late} subsets.

4.2.1 Neural Classifier

The centrepiece of our summarization approach is a classifier separating articles with a strong lead from articles where summary-worthy sentences appear later. We experiment with four

model formulations. A common element of these models is they encode the leading three sentences separately from the overall document and recombine the embeddings to classify the article. The goal of this formulation is to allow the model to learn the interactions between the article's leading sentences and the overall document, in order to properly classify the article. The four models are as follows:

- **Concatenation model:** We separately encode the first three leading sentences and the entire document with BERT. Before encoding, we prepend a [CLS] token to each sentence, then extract the encoded [CLS] token embeddings as sentence representations. We then use separate LSTMs to run over the lead and document sentence embeddings, extracting the final hidden state of both. Shallow MLPs encode these representations to form a *lead representation* $\mathbf{h}_L$ and a *document representation* $\mathbf{h}_D$. Finally the lead and whole document representations are concatenated and a logistic layer assigns a probability of a high proportion score:

$$p\left(\frac{r_{lead}(D)}{r_{oracle}(D)} > T \;\middle|\; \mathbf{h}_L, \mathbf{h}_D\right) = \sigma\left(w_{out}\ [\mathbf{h}_L; \mathbf{h}_D]\right) \tag{4.2}$$

- **Bilinear model:** This model is similar to the concatenation model but instead of concatenating the lead and document representations, a bilinear map combines the embeddings.

$$p\left(\frac{r_{lead}(D)}{r_{oracle}(D)} > T \;\middle|\; \mathbf{h}_L, \mathbf{h}_D\right) = \sigma\left(\mathbf{h}_L^T W_{out}\ \mathbf{h}_D\right) \tag{4.3}$$

- **Separate-BERT model:** This model follows the concatenation model but separate BERT encoders are used to produce the lead and document representations.

- **No-lead baseline:** The leading three sentences are not separately encoded. After BERT encodes the document, the first [CLS] token representation is provided as input to a logistic layer which maps this output to a probability between 0 and 1.

4.2.2 D_{early} and D_{late} Subset Training

After articles are classified, our approach requires two different summarization methods to handle each case. A reasonable strategy is to train separate summarization systems on the D_{early} and D_{late} subsets. We first define a base model based on BanditSum, then detail how to specialize the model on the two cases through subset training.

4.2.2.1 Base Model: BanditSum+BERT

We continue to use BanditSum in these experiments, though we replace the sentence encoder from the original formulation with BERT (Devlin et al. 2019). Given a document D with sentences $s_1, \ldots, s_N$, we encode it using BERT and use the [CLS] token as a sentence embedding, similar to the BertSum summarization model (Liu and Lapata 2019). These sentence embeddings are then fed to a multi-layer perceptron decoder in order to produce a *sentence affinity* $\pi_i \in [0, 1]$ for each sentence $s_i \in D$. Essentially, we compute:

$$\pi_i = \text{MLP}(h_i) \tag{4.4}$$

$$h_1, \ldots, h_N = \text{BERT}(D) \tag{4.5}$$

The MLP structure follows BanditSum's decoder specifications and the remaining model details such as the reinforcement learning-based objective function follow BanditSum's details as well (Dong et al. 2018).

4.2.2.2 Training on Data Subsets

In order to specialize the base model towards summarizing D_{early} and D_{late} articles, we train two separate versions of BanditSum+BERT, restricted to the D_{early} and D_{late} subsets. Our hypothesis is that articles where the lead baseline performs well require a different summarization strategy than cases where the lead is weak. We use BanditSum + $\text{BERT}_{\text{early}}$ to denote the model trained on D_{early} and likewise, use BanditSum + $\text{BERT}_{\text{late}}$ for the model trained on D_{late}.

4.2.3 Summarization through Classification: LeadClassifySum

The final summarization model is a straightforward pipeline of the previous components. Given an article, the classifier predicts if it belongs in D_{early} or D_{late}, and the appropriate model is then used to summarize it. After finding that BanditSum + $\text{BERT}_{\text{early}}$ solely learns positional cues and chooses indices nearly always matching the lead-3 baseline, we instead use the simpler lead-3 baseline to summarize cases in D_{early}. We use BanditSum + $\text{BERT}_{\text{late}}$ to summarize articles in D_{late}. This final model is named LeadClassifySum.

4.3 Experiments

For each of the D_{20}, D_{30}, D_{50} and D_{60} subsets[1], we train the four classification models to separate articles with a proportion score less than $T = 0.75$ from ones greater than T. We also experiment with a setting where no articles are removed, i.e. using D_{all}. Every model is trained to reduce the cross-entropy loss between its prediction and the gold label. For each training subset, we test the model using a subset of the test set that respects the same threshold boundaries as the training subset. We implement data re-weighting to reduce the effects of class imbalance, and we employ gradient clipping with a maximum gradient norm of 1.0.

We train the BanditSum+BERT model on the CNN / Daily Mail dataset, converging after 6 epochs. For both D_{early} and D_{late} subsets, we train separate BanditSum+BERT models, with the late version converging within 10 epochs and the early version within 5 epochs.

For all experiments, we employ the Hugging Face 'bert-base-uncased' BERT implementation to build our models (Wolf et al. 2019). We use Adam to optimize model parameters, with PyTorch's default momentum configuration (Kingma and Ba 2015; Paszke et al. 2019). We tune the learning rate for each of these experiments using the Tune library (Liaw et al. 2018).

[1]See Section 4.1 for how these subsets are defined.

4.4 Results

4.4.1 Classification Results

Results across all model variations are very similar for at least one learning rate configuration. The results for the Bilinear model are shown in Table 4.1. All subsets experience a striking degree of overfitting, with each subset resulting in greater than 20% difference between the training and testing accuracy. In Table 4.2, we compare the Bilinear model to a simple majority baseline. Apart from the D_{all} setting, the classifier outperforms the majority baseline, suggesting that the classifier learns non-trivial cues from the training data. There is also a notable increase in test set accuracy as the amount of data removed increases, with an 11% increase between the D_{all} and D_{60} dataset. However, overall, overfitting prevents our classifier implementations from being practical.

Dataset	Training Accuracy (%)	Test Accuracy (%)
D_{all}	81.64	62.5
D_{20}	90.29	65.09
D_{30}	95.74	67.09
D_{50}	93.31	71.9
D_{60}	93.61	73.49

Table 4.1: Classification results for the Bilinear model on various data subsets, using $T = 0.75$. We notice greater test accuracy as the amount of data removed increases; however, strong overfitting prevents the classifiers from being practical for our purposes.

Model	Accuracy (%)				
	D_{all}	D_{20}	D_{30}	D_{50}	D_{60}
Majority Baseline	64.62	52.61	52.94	53.29	53.16
Bilinear Model	62.5	65.09	67.09	71.9	73.49

Table 4.2: Apart from the D_{all} dataset, the Bilinear model is able to outperform a majority baseline on the classification task.

Model	ROUGE-1	ROUGE-2	ROUGE-L	Lead Overlap (%)
BanditSum+BERT	40.17	17.56	36.61	53.20
BanditSum + $\text{BERT}_{\text{late}}$	40.75	18.17	37.30	33.37
Lead-3	35.32	13.38	31.77	100.0
Oracle	57.19	33.50	53.94	17.13

Table 4.3: Results on CNN / Daily Mail test articles with proportion score less than $T = 0.75$, corresponding to training on D_{late}. The BanditSum + $\text{BERT}_{\text{late}}$ model noticeably outperforms BanditSum+BERT and selects the leading sentences less often.

Model	ROUGE-1	ROUGE-2	ROUGE-L	Lead Overlap (%)
BanditSum+BERT	47.01	22.84	43.18	67.50
BanditSum + $\text{BERT}_{\text{early}}$	49.01	24.52	45.04	99.99
Lead-3	49.01	24.52	45.04	100.0
Oracle	56.16	30.70	52.45	45.70

Table 4.4: Results on CNN / Daily Mail test articles with proportion score less than $T = 0.75$, corresponding to training on D_{early}. Although BanditSum + $\text{BERT}_{\text{early}}$ outperforms BanditSum+BERT significantly, positional cues dominate the learning signal for BanditSum + $\text{BERT}_{\text{early}}$. In the end, it is indistinguishable from the lead-3 baseline.

4.4.2 D_{early} and D_{late} Results

The results for D_{late} and D_{early} are displayed in Tables 4.3 and 4.4 respectively. On D_{late}, subset training benefits the BanditSum+BERT model considerably, as we see large jumps in ROUGE-1, -2 and -L metrics. There is also a strong decrease in the lead overlap, as the BanditSum + $\text{BERT}_{\text{late}}$ model is exposed to less positional bias compared to the baseline model. These results confirm that separating articles with weak lead performance is important for summarization, and that subset training can help models focus on a specific strategy for the D_{late} subset.

The D_{early} experiment shows even greater gains in ROUGE performance, around 2 points across all three ROUGE metrics. However, in this case, positional biases clearly play a dominating role in the model's learning process, as the model exactly copies the lead-3 baseline and achieves the same performance. Unfortunately, the model fails to pick up on deeper cues and its ROUGE scores remain far from the oracle's.

4.4.3 Summarization Results

The main summarization results on the CNN / Daily Mail test set are shown in Table 4.5.

4.4.3.1 BanditSum+BERT Baseline

The BanditSum+BERT markedly outperforms BanditSum across all three ROUGE metrics, indicating that BERT's encodings are far more suitable for this summarization task than LSTMs. Besides a noticeable increase in ROUGE, the model also selects significantly less from the lead than both BanditSum and the BanditSum+KL model. One possible explanation is that BERT acts as a regularizer for sentence position, and its strong language modelling abilities help it from being distracted by position cues.

Experiments with a BanditSum+BERT+KL model do not result in increased ROUGE scores nor decreased lead overlap compared to the BanditSum+BERT model. It is possible that BERT encodings are less reliant than LSTMs on position cues and therefore benefit less from the auxiliary loss regularization. We leave this hypothesis for later experiments and do not investigate the model further.

4.4.3.2 LeadClassifySum

Hampered by a weak classifier, the LeadClassifySum approach is unable to surpass its baseline in terms of ROUGE. Curiously, the degree of lead overlap actually increases, further reflecting on the classifier's inability to properly separate strong-lead articles from weak ones. The summarization approach hinges on a capable classifier, and in the absence of one, this result is expected.

Model	ROUGE-1	ROUGE-2	ROUGE-L	Lead Overlap (%)
Lead-3	40.06	17.53	36.18	100.0
BanditSum	41.68	18.78	38.00	69.87
BanditSum+BERT	42.70	19.40	39.03	57.52
LeadClassifySum (our model)	41.82	18.62	38.12	73.52
Oracle Classifier	43.73	20.49	40.13	52.02
Oracle	56.53	32.65	53.12	27.24

Table 4.5: Results from various models on the CNN / Daily Mail test set. While BanditSum+BERT outperforms BanditSum, both are surpassed by the Oracle Classifier and true Oracle scores. The LeadClassifySum model faces challenges from an inadequate classifier, preventing it from summarizing effectively.

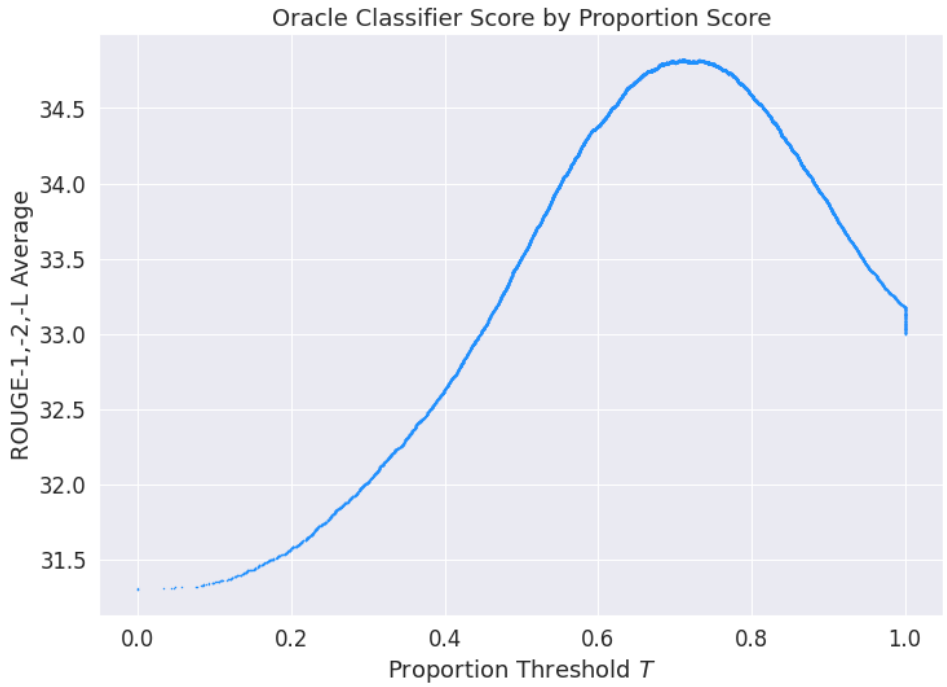

Figure 4.2: By running both BanditSum + $\text{BERT}_{\text{late}}$ and Lead-3 models over the entire test set, we can visualize how the ROUGE-1, -2 and -L average score changes on an article-by-article basis when varying the threshold T. This experiment validates our choice of T and shows that only a single optimum exists.

4.5 Analysis

4.5.1 Exploring Threshold Values

A natural question to ask is what an upper performance bound would be from using this approach. To this end, we run an oracle classifier over the CNN / Daily Mail test set to

correctly label articles with their proportion score. Articles with a proportion score less than T are then summarized with the BanditSum + $\text{BERT}_{\text{late}}$ model. Conversely, articles with a proportion score greater than T are summarized by the lead-3 baseline.

The Oracle Classifier results are displayed in Table 4.5. The noticeable performance gain of over 1 point in ROUGE-1, -2 and -L metrics suggest that our lead classifier approach is a viable approach to summarization. This oracle method is furthermore able to decrease lead overlap by 5.5% compared to the base BanditSum + $\text{BERT}_{\text{late}}$ model, achieving a closer degree of lead overlap as the true oracle model.

We also want to further validate our choice of T. The experiment shown in Fig. 4.1 gives an adequate estimate of a good threshold value. However, we would prefer to use stronger baselines as the base models and analyze the results on a fine-grained article-by-article basis.

We first create summaries for the whole CNN / Daily Mail test set with both lead-3 and BanditSum + $\text{BERT}_{\text{late}}$ models, and sort all articles by their proportion scores. We then sweep through the sorted test set, marking the change in overall average ROUGE-1, -2 and -L scores as T varies. The results are shown in Figure 4.2. The peak ROUGE score, occurring at $T = 0.71$, shows the possible gains over the BanditSum + $\text{BERT}_{\text{late}}$ model (far right point) and lead-3 model (far left point). Figure 4.2 also demonstrates that only a single optimal threshold value exists, and that our choice of T is justified.

4.5.2 Classifier Regularization

In an effort to reduce the degree of overfitting, we explore a variety of regularization techniques to reduce the gap between the training and testing accuracies. We experiment with the following techniques:

- **Weight decay** is a common technique for regularizing parameters which involves multiplying each weight by a factor slightly less than one after each step. We explore a variety of weight decay terms to regularize model parameters.

- **Reducing model complexity:** We halve both the LSTM decoders' hidden size and the logistic classifier size to explore if reducing model size improves generalization.

- **PCA on BERT outputs:** BERT's output representations are 768-dimensional vectors, which may be too large for the task. Using a version of BERT with lower-dimensional outputs requires re-training the entire BERT model from scratch, which is computationally infeasible. Instead, after fine-tuning BERT on the classification task, we fix BERT's parameters and employ PCA to reduce the output dimension to 300. We then continue fine-tuning the decoders using the lower-dimensional outputs.

- **Freeze BERT outputs:** In order to reduce the number of trainable parameters, we fix BERT's parameters, and only fine-tune the remaining weights.

We do not explore using dropout as the BERT encoder already employs it.

Unfortunately, none of the above regularization techniques significantly changes classification accuracy on the test set. Freezing BERT outputs decreases accuracy by around 8%. Further regularization techniques may be beneficial to the task, but ultimately the solution may require alternate model formulations to overcome the gaps in accuracy.

4.6 Conclusion

In this chapter, we conjecture that separating articles with strong or weak leads can aid automatic summarization, though ultimately we do not observe so with the models we build. To partition the dataset, we define a document's *Proportion Score*, the ratio of the lead to oracle ROUGE score. We run experiments using the lead-3 baseline and the BanditSum+KL model to determine an effective threshold T to partition the CNN / Daily Mail dataset.

We design classifiers to divide documents by their proportion score. Despite regularization attempts, overfitting remains a bottleneck for our approach. Future methods may require alternate model formulations or stronger regularization techniques.

We proceed to build a base model using BanditSum with a BERT encoder. After partitioning the training dataset based on proportion score, we train separate summarizers on the subsets. On the late subset, where the documents' proportion scores are less than T, BanditSum + $\text{BERT}_{\text{late}}$ is able to create better summaries, presumably because its representations are less biased towards positional cues. In contrast, BanditSum + $\text{BERT}_{\text{early}}$ summarization performance increases on the early subset, but the model exclusively learns to copy the lead baseline.

Using an oracle classifier, we show that the summarization performance can improve up to 1 point across all ROUGE-1, -2 and -L metrics. We examine how performance gains change on an article-by-article basis, showing that only a single optimum occurs at $T = 0.71$.

5

Conclusion

In this work, we have investigated summarization systems and how positional biases affect these systems' learning process. Motivated by Kedzie, McKeown, and Daume III 2018's work and our own experiments showing that current summarization systems are heavily affected by positional cues, we design novel methods to counter the dominance of these signals.

Our first method involves augmenting an existing gradient descent-based summarization model with an auxiliary loss objective. For a given input article, this loss objective estimates each sentence's value by computing the sentence-level ROUGE score compared to the reference summary. It then encourages the model to match these sentence-value estimates using the KL divergence between the model's predictions and the estimates. We test this method with a state-of-the-art summarization system, BanditSum (Dong et al. 2018), and find that our method can significantly improve summary quality.

Although the improvements are noteworthy, the resulting model is still hampered by an overreliance on lead bias. Evidence for this lingering issue is seen in Figure 3.4, as the oracle summarizer extracts leading sentences far less than the BanditSum+KL method. Some extensions to this method could include other algorithms to combine the loss functions such as MAML (Finn, Abbeel, and Levine 2017), or investigating RL methods aimed at promoting exploration such as the Soft-Actor Critic algorithm (Haarnoja et al. 2018).

After hypothesizing that articles with a strong vs. weak lead require different summarization strategies, we design a second novel summarization method based around classifying

whether an article's lead contains summary-worthy sentences. We find that by partitioning the training dataset and training separate summarization systems on these subsets, we can achieve greater performance on both subsets. We note that in the case of the D_{early} subset, the model's learning is dominated by positional bias, and it learns to mimic the lead-3 baseline exactly. Training a classifier proves to be a much more difficult task, especially with regards to overfitting. Since the classifier does not perform adequately, it remains unclear how to build an effective summarization system using this approach, and our proposal is a negative result for the full summarization system.

Future summarization approaches may be able to indirectly alleviate lead bias issues without explicitly targeting them in their formulation. Emerging summarization models, such as BertSum and BART (Liu and Lapata 2019; Lewis et al. 2019), have taken advantage of large unsupervised language model pre-training in their approaches. In particular, BertSum's authors show that their method is able extract sentences further in the document compared to a competitive baseline, and that the extracted sentence positions are more similar to an oracle summarizer. Compared to previous models such as BanditSum, BertSum is more robust with respect to lead bias effects, despite having no explicit target objective aimed at countering this damaging signal. This may mean that explicit lead bias regularization is not necessary, as long as the model can sufficiently balance positional cues with semantic ones.

Regardless of how future summarization models operate, we have shown that positional biases are an important consideration when designing summarization systems. Creating models that understand how to properly balance positional cues and value sentences correctly represents a major milestone towards truly practical summarization systems.

Bibliography

Berg-Kirkpatrick, Taylor, David Burkett, and Dan Klein (July 2012). "An Empirical Investigation of Statistical Significance in NLP". In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Jeju Island, Korea: Association for Computational Linguistics, pp. 995–1005.

Carbonell, Jaime and Jade Goldstein (1998). "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*. SIGIR '98. Melbourne, Australia: Association for Computing Machinery, pp. 335–336.

Carletta, Jean et al. (2005). "The AMI Meeting Corpus: A Pre-Announcement". In: *Proceedings of the Second International Conference on Machine Learning for Multimodal Interaction*. MLMI'05. Edinburgh, UK: Springer-Verlag, pp. 28–39.

Cheng, Jianpeng and Mirella Lapata (Aug. 2016). "Neural Summarization by Extracting Sentences and Words". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 484–494.

Clarke, James and Mirella Lapata (2008). "Global inference for sentence compression: An integer linear programming approach". In: *Journal of Artificial Intelligence Research* 31, pp. 399–429.

Collobert, Ronan and Jason Weston (2008). "A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning". In: *Proceedings of the 25th International Conference on Machine Learning*. ICML '08. Helsinki, Finland: Association for Computing Machinery, pp. 160–167.

Dang, Hoa Trang (July 2006). "DUC 2005: Evaluation of Question-Focused Summarization Systems". In: *Proceedings of the Workshop on Task-Focused Summarization and Question Answering*. Sydney, Australia: Association for Computational Linguistics, pp. 48–55.

Dang, Hoa Trang and Karolina Owczarzak (2008). "Overview of the TAC 2008 Update Summarization Task." In: *TAC*.

Devlin, Jacob et al. (June 2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186.

Dong, Yue et al. (2018). "BanditSum: Extractive Summarization as a Contextual Bandit". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3739–3748.

Dror, Rotem et al. (July 2018). "The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 1383–1392.

Edmundson, H. P. (Apr. 1969). "New Methods in Automatic Extracting". In: *J. ACM* 16.2, pp. 264–285.

Erkan, G. and D. R. Radev (Dec. 2004). "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization". In: *Journal of Artificial Intelligence Research* 22, pp. 457–479.

Finn, Chelsea, Pieter Abbeel, and Sergey Levine (June 2017). "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In: *Proceedings of the 34th International Confer-*

ence on Machine Learning. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, pp. 1126–1135.

Galashov, Alexandre et al. (2019). "Information asymmetry in KL-regularized RL". In:

Gillick, Dan and Benoit Favre (June 2009). "A Scalable Global Model for Summarization". In: *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*. Boulder, Colorado: Association for Computational Linguistics, pp. 10–18.

Grenander, Matt et al. (Nov. 2019). "Countering the Effects of Lead Bias in News Summarization via Multi-Stage Training and Auxiliary Losses". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 6019–6024.

Grusky, Max, Mor Naaman, and Yoav Artzi (June 2018). "NEWSROOM: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 708–719.

Haarnoja, Tuomas et al. (Oct. 2018). "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, pp. 1861–1870.

Harman, Donna and Paul Over (July 2004). "The Effects of Human Variation in DUC Summarization Evaluation". In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, pp. 10–17.

Hermann, Karl Moritz et al. (2015). "Teaching machines to read and comprehend". In: *Advances in Neural Information Processing Systems*, pp. 1693–1701.

Hong, Kai and Ani Nenkova (2014). "Improving the estimation of word importance for news multi-document summarization". In: *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 712–721.

Kågebäck, Mikael et al. (Apr. 2014). "Extractive Summarization using Continuous Vector Space Models". In: *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*. Gothenburg, Sweden: Association for Computational Linguistics, pp. 31–39.

Kedzie, Chris, Kathleen McKeown, and Hal Daume III (2018). "Content Selection in Deep Learning Models of Summarization". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1818–1828.

Kingma, Diederik P and Jimmy Ba (2015). "Adam: A method for stochastic optimization". In: *International Conference for Learning Representations (ICLR)*.

Kornilova, Anastassia and Vladimir Eidelman (Nov. 2019). "BillSum: A Corpus for Automatic Summarization of US Legislation". In: *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. Hong Kong, China: Association for Computational Linguistics, pp. 48–56.

Kulesza, Alex, Ben Taskar, et al. (2012). "Determinantal point processes for machine learning". In: *Foundations and Trends® in Machine Learning* 5.2–3, pp. 123–286.

Lewis, Mike et al. (2019). *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*.

Liaw, Richard et al. (2018). "Tune: A Research Platform for Distributed Model Selection and Training". In: *arXiv preprint arXiv:1807.05118*.

Lin, Chin-Yew (July 2004). "ROUGE: A Package for Automatic Evaluation of Summaries". In: *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*. Ed. by Stan Szpakowicz Marie-Francine Moens.

Lin, Hui and Jeff Bilmes (June 2011). "A Class of Submodular Functions for Document Summarization". In: *Proceedings of the 49th Annual Meeting of the Association for Compu-*

tational Linguistics: Human Language Technologies. Portland, Oregon, USA: Association for Computational Linguistics, pp. 510–520.

Liu, Yang and Mirella Lapata (Nov. 2019). "Text Summarization with Pretrained Encoders". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 3730–3740.

Luhn, Hans Peter (1958). "The Automatic Creation of Literature Abstracts". In: *IBM Journal of Research and Development* 2, pp. 159–165.

McDonald, Ryan (2007). "A study of global inference algorithms in multi-document summarization". In: *European Conference on Information Retrieval*. Springer, pp. 557–564.

Mihalcea, Rada and Paul Tarau (July 2004). "TextRank: Bringing Order into Text". In: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain: Association for Computational Linguistics, pp. 404–411.

Mikolov, Tomas et al. (2013). "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems*, pp. 3111–3119.

Nachum, Ofir, Mohammad Norouzi, and Dale Schuurmans (2017). "Improving policy gradient by exploring under-appreciated rewards". In: *International Conference on Learning Representations*, pp. 2775–2785.

Nallapati, Ramesh, Feifei Zhai, and Bowen Zhou (2017). "SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents". In: *Proceedings of the 31st AAAI Conference on Artificial Intelligence*.

Nallapati, Ramesh, Bowen Zhou, et al. (2016). "Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond". In: *Conference on Computational Natural Language Learning (CoNLL)*, pp. 280–290.

Napoles, Courtney, Matthew Gormley, and Benjamin Van Durme (June 2012). "Annotated Gigaword". In: *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*. Montréal, Canada: Association for Computational Linguistics, pp. 95–100.

Narayan, Shashi, Shay B. Cohen, and Mirella Lapata (2018a). "Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium.

— (2018b). "Ranking Sentences for Extractive Summarization with Reinforcement Learning". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*.

Nenkova, Ani (2005). "Automatic text summarization of newswire: Lessons learned from the document understanding conference". In: *AAAI*. Vol. 5, pp. 1436–1441.

Nenkova, Ani and Lucy Vanderwende (2005). "The impact of frequency on summarization". In: *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005* 101.

Paszke, Adam et al. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., pp. 8024–8035.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

PyTables Developers Team (2002-2019). *PyTables: Hierarchical Datasets in Python*.

Radev, Dragomir R., Hongyan Jing, and Malgorzata Budzikowska (2000). "Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies". In: *NAACL-ANLP 2000 Workshop: Automatic Summarization*.

Rennie, Steven J et al. (2017). "Self-Critical Sequence Training for Image Captioning". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7008–7024.

Sandhaus, Evan (2008). *The New York Times Annotated Corpus LDC2008T19*. DVD. Philadelphia: Linguistic Data Consortium.

Schiffman, Barry, Ani Nenkova, and Kathleen McKeown (2002). "Experiments in Multidocument Summarization". In: *Proceedings of the Second International Conference on Human Language Technology Research*, pp. 52–58.

Sharma, Eva, Chen Li, and Lu Wang (2019). "BIGPATENT: A Large-Scale Dataset for Abstractive and Coherent Summarization". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

The HDF Group (1997-2019). *Hierarchical Data Format, version 5*. http://www.hdfgroup.org/HDF5/.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in Neural Information Processing Systems*, pp. 6000–6010.

Williams, Ronald J (1992). "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Reinforcement Learning*. Springer, pp. 5–32.

Wolf, Thomas et al. (2019). "HuggingFace's Transformers: State-of-the-art Natural Language Processing". In: *ArXiv* abs/1910.03771.

Wu, Yuxiang and Baotian Hu (2018). "Learning to Extract Coherent Summary via Deep Reinforcement Learning". In: *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*. Ed. by Sheila A. McIlraith and Kilian Q. Weinberger.

Zhang, Yuhao et al. (2018). "Learning to Summarize Radiology Findings". In: *EMNLP 2018 Workshop on Health Text Mining and Information Analysis*.

Zhou, Qingyu et al. (July 2018). "Neural Document Summarization by Jointly Learning to Score and Select Sentences". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 654–663.