4-29-2021 10:30 AM

Improving Reader Motivation with Machine Learning

Tanner A. Bohn, *The University of Western Ontario*

Supervisor: Ling, Charles X., *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree in Computer Science

Abstract

This thesis focuses on the problem of increasing reading motivation with machine learning (ML). The act of reading is central to modern human life, and there is much to be gained by improving the reading experience. For example, the internal reading motivation of students, especially their interest and enjoyment in reading, are important factors in their academic success.

There are many topics in natural language processing (NLP) which can be applied to improving the reading experience in terms of readability, comprehension, reading speed, motivation, etc. Such topics include personalized recommendation, headline optimization, text simplification, and many others. However, to the best of our knowledge, this is the first work to explicitly address the broad and meaningful impact that NLP and ML can have on the reading experience.

In particular, the aim of this thesis is to explore new approaches to supporting internal reading motivation, which is influenced by readability, situational interest, and personal interest. This is performed by identifying new or existing NLP tasks which can address reader motivation, designing novel machine learning approaches to perform these tasks, and evaluating and examining these approaches to determine what they can teach us about the factors of reader motivation.

In executing this research, we make use of concepts from NLP such as textual coherence, interestingness, and summarization. We additionally use techniques from ML including supervised and self-supervised learning, deep neural networks, and sentence embeddings.

This thesis, presented in an integrated-article format, contains three core contributions among its three articles. In the first article, we propose a flexible and insightful approach to coherence estimation. This approach uses a new sentence embedding which reflects predicted position distributions. Second, we introduce the new task of pull quote selection, examining a spectrum of approaches in depth. This article identifies several concrete heuristics for finding interesting sentences, both expected and unexpected. Third, we introduce a new interactive summarization task called HARE (**H**one **a**s You **Re**ad), which is especially suitable for mobile devices. Quantitative and qualitative analysis support the practicality and potential usefulness of this new type of summarization.

Keywords: Natural language processing, machine learning, internal motivation, reading, coherence, sentence embedding, attention, situational interest, pull quotes, interactive summarization

Lay Summary

Reading is an increasingly important human skill. The interest and enjoyment students have in reading for example is an important factor in their academic success. This thesis is concerned with how to apply techniques from machine learning (ML) and natural language processing (NLP) in order to improve how readable, attention grabbing, or personally relevant reading material is, especially in a digital setting. ML allows us to automatically identify patterns and trends in large datasets, and NLP is concerned with the application of computer science to naturally occurring language, such as news articles.

In this thesis, we consider three NLP problems which are related to reader enjoyment and interest, and we propose new solutions to those problems. The first problem we consider is related to determining the readability of a text based on how well its concepts are organized (a property known as coherence). The solution we propose works by learning to look at each sentence out of context and predicting where it should belong. Second, we propose a new problem called pull quote (PQ) selection. PQs are often found in newspapers or online news articles, and are sentences or quotations from the article placed in an eye-catching graphic. They are intended to grab the reader's attention and make them interested in reading more of the article. We propose several methods for learning to choose good PQs from a text, and learn about unexpected properties of PQs in the process. Third, we introduce a new type of reading assistance tool suitable for mobile devices. This tool is based on the NLP problem of interactive personalized summarization, and is intended to use low-effort feedback during reading to understand reader preferences and provide them with personalized summaries. We propose several approaches capable of predicting what parts of an article they will be interested in reading and demonstrate the practicality of this type of tool.

Aside from topics in NLP, research completed during the course of this PhD (but not included in thesis) touched on abstract visual reasoning problems and lifelong machine learning (learning many tasks in sequence, especially without forgetting earlier tasks).

Co-Authorship Statement

I acknowledge my supervisor, Charles Ling, as a coauthor on the three integrated articles. Additionally, Yining Hu and Jinhang Zhang contributed to coding and evaluating baselines for the first integrated article “Learning Sentence Embeddings for Coherence Modelling and Beyond”. Tanner Bohn wrote the remainder of the code for the first paper, as well as all necessary coding for the last two integrated articles. Tanner also performed all necessary experiment design, analysis of results, and writing, under the supervision of Dr. Ling.

Publication statuses of the integrated articles:
1. “Learning Sentence Embeddings for Coherence Modelling and Beyond”: accepted to the International Conference on Recent Advances in Natural Language Processing (RANLP) 2019 and published by the Association for Computational Linguistics (ACL);
2. “Catching Attention with Automatic Pull Quote Selection”: accepted to the International Conference on Computational Linguistics (COLING) 2020 and published by ACL;
3. “Hone as You Read: A Practical Type of Interactive Summarization”: Submitted to the Conference of the Association for Computational Linguistics 2021.

Acknowlegements

A PhD is an undertaking best not done alone, and I have many people to thank both for inspiring me to go on this journey as well as helping me along the way. First, I would like to thank my PhD supervisor Charles Ling. The freedom you gave me to investigate and follow my curiosity and interests has been greatly appreciated. Your many wise words on performing meaningful research and the importance of prioritization were invaluable.

I want to thank my many colleagues and friends at Western, both past and present, who provided words of encouragement, inspiring discussion, and fun game nights. In particular, I would like to thank Xinyu Yun, who was always available to chat in the lab and who worked through many stressful times and paper deadlines with me.

For making the technical aspects of my graduate experience remarkably smooth, I am very grateful for the departmental members, especially Janice Wiersma and Dianne McFadzean. Thank you.

I also owe my gratitude to the many passionate educators and mentors I have had on my long educational journey. You have all helped me grow, challenge myself, and discover my passions.

I would like to thank my partner, Marina, who has been endlessly encouraging and patient, and who served as a sounding board for many research ideas. Finally, I am grateful for the unyielding support of my family. To my parents, Tom and Glenda, thank you.

Contents

Abstract ii

Lay Summary iii

Co-Authorship Statement iv

Acknowlegements v

List of Figures viii

List of Tables xi

1 Introduction 1

1.1 Motivation 1

1.1.1 Importance of Improving the Reading Experience 1

1.1.2 How the Reading Experience Can Be Improved 2

1.2 Research Aim and Objectives 4

1.3 Contributions 4

1.4 Thesis Format 5

1.5 Publications and Preprints 6

2 Background 7

2.1 Coherence 7

2.2 Situational Interest 8

2.3 Document Summarization 10

2.4 Fundamental Machine Learning Concepts 12

2.4.1 Learning Paradigms 12

2.4.2 Central Challenges of Machine Learning 13

2.5 Artificial Neural Networks 14

2.5.1 Basics 14

2.5.2 Regularization 17

2.5.3 Important Types of Neural Networks 18

2.6 Text Embeddings 20

2.6.1 Handcrafted Features 20

2.6.2 n-Gram Representations 20

2.6.3 Distributed Embeddings 21

3 Paper 1 **23**

4 Paper 2 **34**

5 Paper 3 **51**

6 Conclusion **64**

6.1 Synopsis 64

6.2 Summary of Findings 65

6.3 Future Directions 66

Bibliography **68**

Curriculum Vitae **77**

List of Figures

1.1 Several aspects of the reading experience, as well as the relevance of many NLP tasks. The NLP tasks that may be used to influence a given aspect of reading are indicated with links. The three NLP tasks which are the focus of this thesis are highlighted in blue: pull quote selection (a new task), automated summarization (we propose a particular new type of personalized summarization), and coherence modelling. 3

1.2 Contributions in this work. 4

2.1 An example of a PQ from `https://theintercept.com/2019/12/05/us-visa-social-media-sc`

2.2 Examples of underfitting, optimal fitting, and overfitting in regression (supervised learning) and clustering (unsupervised learning). We consider a clustering model to overfit when it identifies structures in the data, such as additional "groups", that are likely the result of noise. 13

2.3 A perceptron, showing the input $(x_0, x_1, x_2, ...)$, the weights $(w_0, w_1, w_2, ...)$, and the bias, b. 15

2.4 Multi-layer neural network (biases not shown). 16

2.5 A simplified diagram of a single-layer recurrent network. At each time step, it receives the next element of the input sequence as well as the layer output from the previous time step. 19

2.6 Converting a sentence into various word and character level n-grams. 20

2.7 Skip-gram architecture with a context size of 2 (two words before and after an input word). By training a neural network to predict the words surrounding an input word, we can learn useful distributed word representations. 22

3.1 This paper abstract is analyzed by our sentence position model trained on academic abstracts. The sentence encodings (predicted position distributions) are shown below each sentence, where white is low probability and red is high. Position quantiles are ordered from left to right. The first sentence, for example, is typical of the first sentence of abstracts as reflected in the high first-quantile value. For two text coherence tasks, we show the how the sentence encodings can easily be used to solve them. The black dots indicate the weighted average predicted position for each sentence. 24

3.2 Illustration of the sentence position model, consisting of stacked BiLSTMs. Sentences from a text are individually fed into the model to produce a PPD sequence. In this diagram we see a word sequence of length three fed into the model, which will output a single row in the PPD sequence. 26

3.3 A visualization of our NLP algorithms utilizing PPDs applied to a news article. To reorder sentences, we calculate average weighted positions (identified with black circles) to induce an ordering. Coherence is calculated with the Kendall's rank correlation coefficient between the true and induced ranking. We also show how PPDs can be used to perform summarization, as we will explore further in Section 4. 27

3.4 The PPDs for a CNN article. (full text available at http://web.archive.org/web/20150801040019id_/http://www.cnn.com/2015/03/13/us/tulane-bacteria-exposure/). The dashed line shows the weighted average predicted sentence positions. 29

4.1 A pull quote from this paperchosen with the help of our best performing model (see Section 5.3). 35

4.2 Factors suggested by our results to be important (and unimportant) in creating pull quotes. 36

4.3 The progression of neural network architectures combined with SBERT sentence and document embeddings. Group A only uses sentence embeddings, while groups B and C also use document embeddings. In group C, they are combined in a mixture-of-experts fashion (the width of the sigmoid and softmax layers is equal to the # experts). For each group, there is a basic version and deep version. 39

4.4 The value distributions for two interesting handcrafted features for both non-PQ sentences (solid blue region) and PQ sentences (dashed orange lines). . . . 41

4.5 The ten highest and lowest weighted n-grams for the best character and word models. 42

4.6 The top five best performing SBERT embedding dimensions, along with the terms associated with increasing PQ probability with respect to that dimension. For each dimension, we also include the sentence from the test articles which that dimension most strongly scores as being a PQ sentence. At the top of each box is the dimension index and the test AUC_{avg}. 42

5.1 Potential feedback methods for HARE used on a smartphone. In (a), users can choose to swipe left or right to indicate interest or disinterest in sections of text as they read. Users may also provide implicit feedback in the form of dwell time in center window (b) or gaze location, as measured by camera for example (c). More interesting text may have longer gazes or dwell time. The approaches evaluated in this paper rely on feedback similar to (a), but further development in HARE can extend to (b) or (c). 52

5.2 In HARE, users are shown sentences in their original order, and can provide relevance feedback. A model uses this feedback to optimize the remainder of the article, automatically hiding uninteresting sentences. 54

5.3 The performance for HideNextSimilar and HideAllSimilar for a range of similarity thresholds. When the threshold is high, it means that only the most similar sentences are hidden. 58

5.4 Summary of human evaluation results. Error bars indicate 90% confidence intervals. 59
5.5 A screenshot of the demo in action. For each sentence, users were able to accept, reject, or stop reading the article at that point. 63

List of Tables

3.1 The neural sentence position model hyperparameters used in our coherence experiments. The following settings are used across all tasks: batch size of 32, sentence trimming/padding to a length of 25 words, the vocabulary is set to the 1000 most frequent words in the associated training set. The Adamax optimizer is used [54] with default parameters supplied by Keras [14] 28

3.2 Results on the order discrimination and sentence reordering coherence tasks. Our approach trades only a small decrease in performance for improved utility of the sentence embeddings over other approaches, achieving close to or the same as the state-of-the-art. 29

3.3 ROUGE scores on the CNN/DailyMail summarization task. Our PPD-based heuristic outperforms the suite of established heuristic summarizers. However, the higher performance of the deep-learning models demonstrates that training explicitly for summarization is beneficial. 30

4.1 Statistics of our PQ dataset, composed of articles from four different news outlets. Only articles with at least one PQ are included in the dataset. 40

4.2 AUC_{avg} scores of the n-gram models. 42

4.3 Results on the neural architectures. Performance mean and std. dev. is calculated with five trials. k refers to the # experts, only applicable to C group models. Width values correspond to the width of the two additional fully connected layers (only applicable to the deep models). 42

4.4 Performance of the cross-task models. 43

4.5 The results of human evaluation comparing models in terms of how interested the reader is in reading more of the article. The ↑ and ↓ indicate whether better values for a metric are respectively higher or lower. 43

4.6 Hyperparameter values used in grid search for the different SBERT neural networks. The models with the best performance on the validation set averaged across 5 trials are reported in Table 3. 47

4.7 Article source: `https://nationalpost.com/news/politics/federal-carbon-tax-could-degrade-canadian-fishing-industrys-competitiveness-says-memo`. 48

4.8 Article source: `https://www.cosmopolitan.com/lifestyle/a20107039/south-carolina-prom-king-adam-bell-interview/` 48

4.9 Article source: `https://ottawacitizen.com/opinion/columnists/armchair-mayor-fewer-cars-more-transit-options-would-invigorate-ottawa` 49

4.10 Article source: `https://theintercept.com/2019/01/09/trump-speech-democratic-response/`. This article demonstrates a case where there are many real PQs in an article. It also highlights the need for future work which can create multi-sentence PQs (True PQ #4 consists of two sentences). 49

5.1 A comparison of each model proposed. For parameterized models, results with the best variation are reported (for all models, we found that the same parameters performed best for both $score_{sharp}$ and $score_{noisy}$). Non-deterministic models are marked by a *. $score_{adv}$ is the difference between $score_{noisy}$ and the control score (which is independent of feedback). 57

5.2 Results for the two LR model version. For the constant-ϵ variation, a greater ϵ indicates greater exploration. For the decreasing-ϵ variation, larger β indicates a faster decay in exploration probability. 58

5.3 Results for the first two simple heuristic models. For ShowModulo, every k^{th} sentence is shown. For HideNext, the n sentences following a swiped one are hidden. 62

5.4 Results for the two variations of adapted generic summarizer models, for each of three extractive summarizers tested. For GenFixed, $frac$ indicates what fraction of the document is shown, after first sorting sentences by importance. For GenDynamic, ϵ is used for ϵ-greedy exploration to estimate length preference. . 62

5.5 Results for the CoverageOpt model. c controls the initial estimate for concept importances and β controls how smoothly a concept shifts between important and unimportant. 63

Chapter 1

Introduction

1.1 Motivation

The ultimate goal of this thesis is to research novel computational approaches for improving the reading experience. This is an experience that humans have now been engaged in for over 5000 years (since 3200 BC). The first readers, the Sumerians of Mesopotamia (present-day Iraq), used a writing system known as cuneiform, which consisted of simple shapes impressed on clay tablets [31, 96]. As far as we know, the original uses of written language were utilitarian (e.g., recording financial transactions). However, by about 2500 BC, reading and writing had developed to be capable of capturing and conveying almost any thought in the Sumerian language [31, 96].

The presence of writing and the act of reading is now ubiquitous in modern life, taking on all forms and contexts. We read for work, enjoyment, education, and ritual. We read road signs and receipts. We read on paper, plastic, smartphones and smartwatches. We read everywhere for all reasons. Reading is so central to our lives that literacy literally changes the organization of our brains [20].

1.1.1 Importance of Improving the Reading Experience

As a function of its ubiquity, there is much to be gained by improving the reading experience. Consider two application areas:

- **Early education.** Developing the reading comprehension skills of students is an important problem in early education [69, 101, 53]. Several motivations for development in this area were discussed by Catherine Snow almost two decades ago [101], with many of the reasons still being applicable today. Two reasons are unacceptable gaps in reading performance between children of different demographics and high school graduates facing an increasing need for a high degree of literacy, but their education not keeping pace with this increase. If the reading experience can be improved to increase student motivation for example, then an increase of comprehension skills is expected to follow [69]. In this way, by making reading more enticing, enjoyable, or interesting for students, they, and the encompassing society, benefit in both the short and long term via a more educated population.

- **News outlets.** Readership and revenue from printed newspapers has been steadily declining in recent decades [46]. Increasingly, online news media sites are a primary source of news [11]. In this ever more competitive online environment, there are a multitude of ways a news site can improve traffic. While simply increasing site loading speed is an effective and early discovered method [46], there are several proven methods which rely on improving the reading experience. This includes improving the quality of the writing (especially by increasing the quality of the lede at the very beginning of a news article), producing more condensed, less "bloated" content, personalizing the content recommended to users, optimizing headlines, and adding images and videos to accompany the text [46]. When it comes to optimizing online news content, care must be taken. For example, by optimizing headlines solely to maximize clicks, the result is so-called "clickbait", which can misrepresent the article and over time damage the newspaper's brand and reader trust [12, 46].

1.1.2 How the Reading Experience Can Be Improved

The reading experience is highly multifaceted. In studying how to improve the reading experience, there are many interconnected and overlapping directions one could consider. These include reading speed [10], quality of comprehension [69], text readability [25], internal motivation factors such as enjoyment [21] and interestingness [32], and factors of extrinsic motivation such as deadlines, tests, and social aspects [108]. For each of these directions, Figure 1.1 suggests relevant areas of natural language processing (NLP).

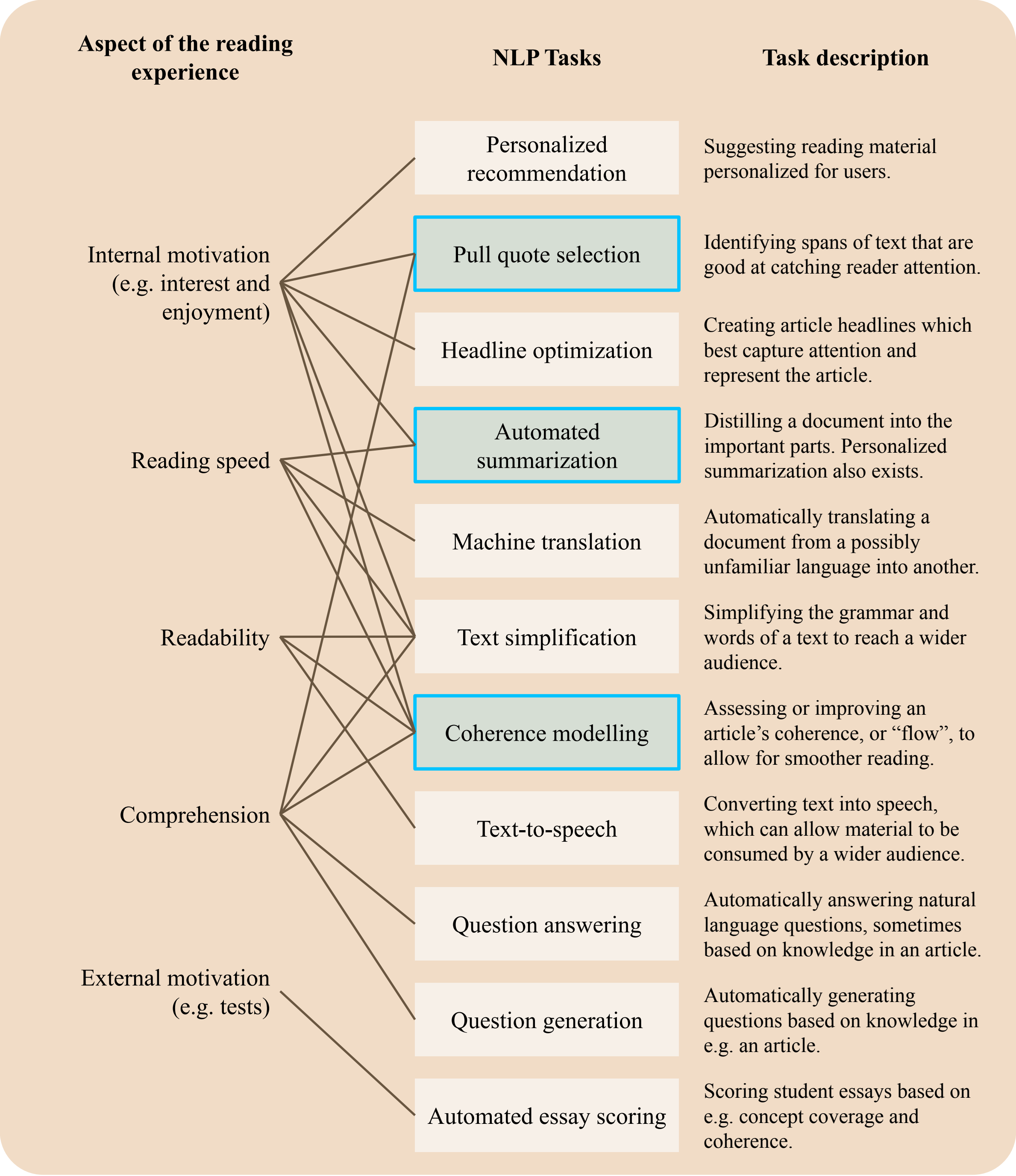
The work in this thesis focuses on the two internal motivation factors of enjoyment and interest in particular. Several factors, including the following, have been found to influence enjoyment and interest:

- **Reading difficulty.** Ease of comprehension, affected by such factors as vocabulary, organization, coherence, and grammar are important in supporting reader interest [97, 25]. The readability of social media posts has also been found to influence engagement, with simpler texts having more likes, comments, and shares [84].

- **Situational interest.** Interest in a text can be broken down into situational interest and personal (or "topic") interest [32]. Situational interest is short-lived and depends on the reading context and presentation of the text. It tends to be a result of novelty, curiosity, and information saliency. A particular text tends to evoke a similar situational interest level across individuals.

- **Personal interest.** This type of interest is generally stable over time and unique to individuals, influenced by their personal experiences and knowledge, and exists prior to encountering a particular text. Consequently, ensuring personal relevance of reading materials has been found to support student engagement and motivation [2, 41].

Next, we will outline how the goals of this thesis are aligned with these three aspects of internal motivation.


Aspect of the reading experience
Internal motivation (e.g. interest and enjoyment)
Reading speed
Readability
Comprehension
External motivation (e.g. tests)

NLP Tasks	Task description
Personalized recommendation	Suggesting reading material personalized for users.
Pull quote selection	Identifying spans of text that are good at catching reader attention.
Headline optimization	Creating article headlines which best capture attention and represent the article.
Automated summarization	Distilling a document into the important parts. Personalized summarization also exists.
Machine translation	Automatically translating a document from a possibly unfamiliar language into another.
Text simplification	Simplifying the grammar and words of a text to reach a wider audience.
Coherence modelling	Assessing or improving an article's coherence, or "flow", to allow for smoother reading.
Text-to-speech	Converting text into speech, which can allow material to be consumed by a wider audience.
Question answering	Automatically answering natural language questions, sometimes based on knowledge in an article.
Question generation	Automatically generating questions based on knowledge in e.g. an article.
Automated essay scoring	Scoring student essays based on e.g. concept coverage and coherence.

Figure 1.1: Several aspects of the reading experience, as well as the relevance of many NLP tasks. The NLP tasks that may be used to influence a given aspect of reading are indicated with links. The three NLP tasks which are the focus of this thesis are highlighted in blue: pull quote selection (a new task), automated summarization (we propose a particular new type of personalized summarization), and coherence modelling.

1.2 Research Aim and Objectives

Aim The aim of this thesis is to investigate novel approaches to supporting internal reading motivation using machine learning. In particular, we will aim to apply concepts from machine learning and the overlapping field of natural language processing (NLP) to improving text readability, situational interestingness, and personal interestingness.

Objectives In order to achieve the aim, we will:

1. identify existing or new tasks in NLP whose solutions can be used to address reader motivation;
2. devise novel machine learning approaches to these tasks;
3. and evaluate and examine these approaches to determine what they can teach us about the various factors of reader motivation.

To lend concreteness to these objectives, the particular tasks we consider are 1) coherence modelling (related to readability), 2) a new task called pull quote (PQ) selection (related to situational interest), and 3) a new task called **H**one **a**s You **Re**ad (HARE) (related to personal interest).

1.3 Contributions

The research contained in this thesis will focus on the three described factors affecting internal motivation of readers: coherence, situational interestingness, and personal interestingness. We show how the contributions in this thesis relate to these three factors in Figure 1.2.

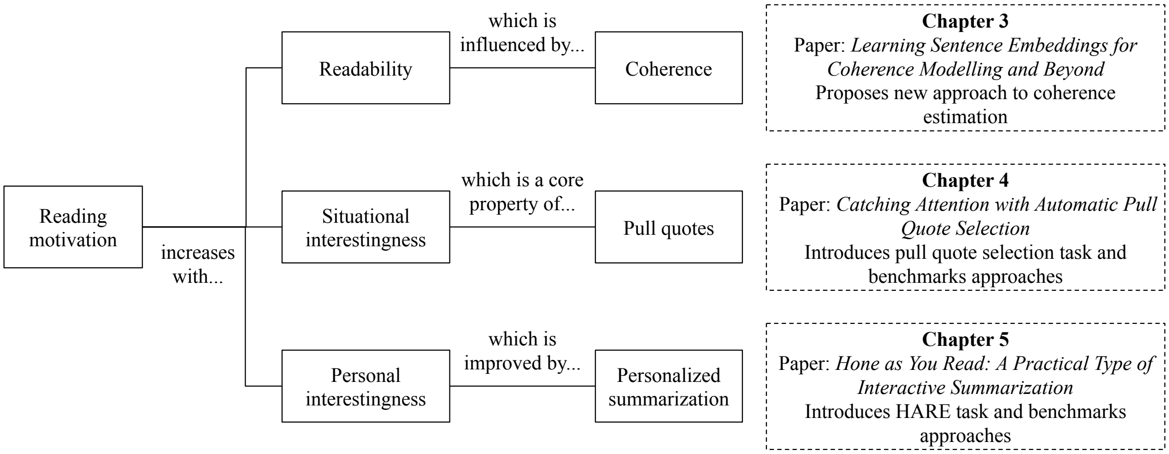

Reading motivation
increases with...
Readability
which is influenced by...
Coherence
Situational interestingness
which is a core property of...
Pull quotes
Personal interestingness
which is improved by...
Personalized summarization
Chapter 3
Paper: Learning Sentence Embeddings for Coherence Modelling and Beyond
Proposes new approach to coherence estimation
Chapter 4
Paper: Catching Attention with Automatic Pull Quote Selection
Introduces pull quote selection task and benchmarks approaches
Chapter 5
Paper: Hone as You Read: A Practical Type of Interactive Summarization
Introduces HARE task and benchmarks approaches

Figure 1.2: Contributions in this work.

The contributions of this thesis can be classified into three types: the proposal of new tasks, the development of new algorithms and techniques, and empirical results.

In terms of new tasks, we expand the reach of NLP into the reading experience by:

- proposing the new task of automatic PQ selection and constructing a dataset for training and evaluation of models for this task (Chapter 4);
- and we define the novel HARE task, describing a suitable evaluation technique to accompany it (Chapter 5).

In terms of algorithmic contributions,

- we propose a novel self-supervised approach to learn sentence embeddings, which we call predicted position distributions (PPDs) (Chapter 3);
- we describe how PPDs can be applied to established coherence tasks using simple algorithms amenable to visual approximation (Chapter 3);
- we describe several motivated approaches for the new task of PQ selection, including a mixture-of-experts approach to combine sentence and document embeddings (Chapter 4);
- and we describe several motivated approaches for HARE, ranging from simple heuristics to adapted generic summarizers, to interest-learning approaches (Chapter 5).

Finally, in terms of empirical contributions,

- we demonstrate that PPDs are competitive at solving text coherence tasks while quickly providing access to further insights into organization of texts (Chapter 3);
- we inspect the performance of our PQ-selection approaches to gain a deeper understanding of PQs, their relation to other tasks, and what engages readers (Chapter 4);
- and we evaluate our HARE approaches to gain a deeper understanding of the task (Chapter 5).

1.4 Thesis Format

This thesis is presented in the integrated-article format. In Chapter 2 we introduce key concepts needed to understand our objectives and their solutions. This chapter covers topics in NLP as well as machine learning. Chapters 3, 4, and 5 are integrated articles from relevant projects completed during the duration of the author's PhD. They represent the three prongs of this thesis. Each of these chapters first provides the publication status of the article and places it in the context of this thesis. Each article, by nature of being a self-contained paper, contains a discussion of more domain-specific motivations, a small literature review, and detailed descriptions of the novel work and experimental results. Chapter 6 concludes the thesis and discusses many possible future directions.

1.5 Publications and Preprints

The work in this thesis is directly related to the following three articles, listed in the order of publication (and expected publication, in the case of the third article which is still under review):

1. **Tanner Bohn**, Yining Hu, Jinhang Zhang, and Charles Ling. Learning sentence embeddings for coherence modelling and beyond. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 151–160, Varna, Bulgaria, September 2019. INCOMA Ltd.

2. **Tanner Bohn** and Charles Ling. Catching attention with automatic pull quote selection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 62–76, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

3. **Tanner Bohn** and Charles Ling. Hone as you read: A practical type of interactive summarization. Submitted to the Conference of the Association for Computational Linguistics, 2021.

While not directly related, the following articles have also been completed over the course of the PhD:

4. **Tanner Bohn**, Yining Hu, and Charles X. Ling. Few-shot abstract visual reasoning with spectral features. *arXiv preprint arXiv:1910.01833*, 2019.

5. Xinyu Yun, **Tanner Bohn**, & Charles X. Ling. (2020, May). A Deeper Look at Bongard Problems. In *Canadian Conference on Artificial Intelligence* (pp. 528-539). Springer, Cham.

6. Xinyu Yun, **Tanner Bohn**, and Charles X. Ling. Tackling Non-forgetting and Forward Transfer with a Unified Lifelong Learning Approach. Appeared at the *4th Lifelong Learning Workshop at ICML*, 2020.

7. Charles X. Ling and **Tanner Bohn**. A Conceptual Framework for Lifelong Learning. *arXiv preprint arXiv:1911.09704*, 2020.

Chapter 2

Background

In this chapter we will briefly cover the concepts most important to understanding the contents of this thesis. In the first three sections, we will introduce concepts from NLP, namely textual coherence, situational interestingness, and summarization. In the next three sections, we will introduce basic machine learning concepts used throughout the work in this thesis, neural networks, and sentence embeddings.

2.1 Coherence

Coherence is an organizational property of text, where better-organized texts are more coherent, making them easier to read and comprehend [58, 86]. As coherence is a property of readability, coherence modelling has found many applications. Examples include refinement of multi-document summaries [3], automatic scoring of essay quality [30], the detection of schizophrenia (through analyzing coherence of speech) [27], and machine translation [56]. Within a given text, the coherence is often considered at occurring at two scales:

Local coherence This refers to the relatedness of text at the sentence-to-sentence transition level. For example, the sentence pair “It is pleasant outside. Sally will go for a walk.” is coherent because the fact that it is pleasant outside is easily understood as the reason that Sally is going for a walk. However, if the pair was “It is pleasant outside. Sally has a brother.”, then this would not be coherent. The evaluation of local coherence models can be performed with the discrimination of documents with locally-permuted sentences [77]. That is, the ability of the model to distinguish between two versions of the same document: one where all sentences are in the correct order, and one where only a few contiguous sentences are shuffled. The task of producing a coherent ordering of a set of sentences can also be considered to evaluate local coherence models [65]. A models can produce such a global ordering by maximizing local coherence throughout the document.

Global coherence This refers to the higher-order structuring of the text. For example, a coherent news article will often begin with a group of sentences describing a central event. This is naturally followed by reporting the cause of the event, then the effect, and then background information [117]. If these groups are not well-separated or occur out of order, the document may lack global coherence, even if the sentences locally occur in a mostly-coherent fashion. The evaluation of global coherence models can be performed with the discrimination of documents with globally-permuted sentences [4, 77]. That is, distinguishing between the original document and one where all sentences are shuffled.

In this thesis, coherence is most relevant to Chapter 3, where we describe a new method for estimating coherence, sensitive to both local and global coherence. In the associated article, we also further discuss previous approaches to these tasks.

2.2 Situational Interest

While coherence may influence the readability of a text and the internal motivation of readers while they read, it is *situational interest* which often incites the motivation for them to begin reading[1].

Situational interest appears in the form of spontaneous curiosity and an attraction to novelty or salient information. This type of interest (in contrast to *topic interest* which features in Section 2.3) is short-lived and depends on the context [32]. For an example of such context-dependent interest, consider a student tasked with learning about the Battle of Vimy Ridge; they may have no pre-existing interest in the topic, but in the context of completing their assignment, any article on the topic may catch their attention. Understanding and predicting what we find interesting, attention-grabbing, and appealing has been studied for a wide variety of content types and domains such as music [62], images and video [23, 89], web-page aesthetics [91], as well as online news article content [57, 19].

The situational interest of documents, particularly online content, can be increased through many means. An online news outlet, wishing to increase the attractiveness of their articles, may including salient images or videos for example. Emotionally significant visual media in particular has been shown to capture attention [98], and the task of automatically predicting interestingness of images and video scenes has also been studied [23, 89]. To capitalize on novelty, stories regarding recent events should be published, and to further ensure contextual interest, stories originating from near the target audience should be preferred [26] (e.g., an article about election results in a target country are generally most interesting to residents of that country around election time).

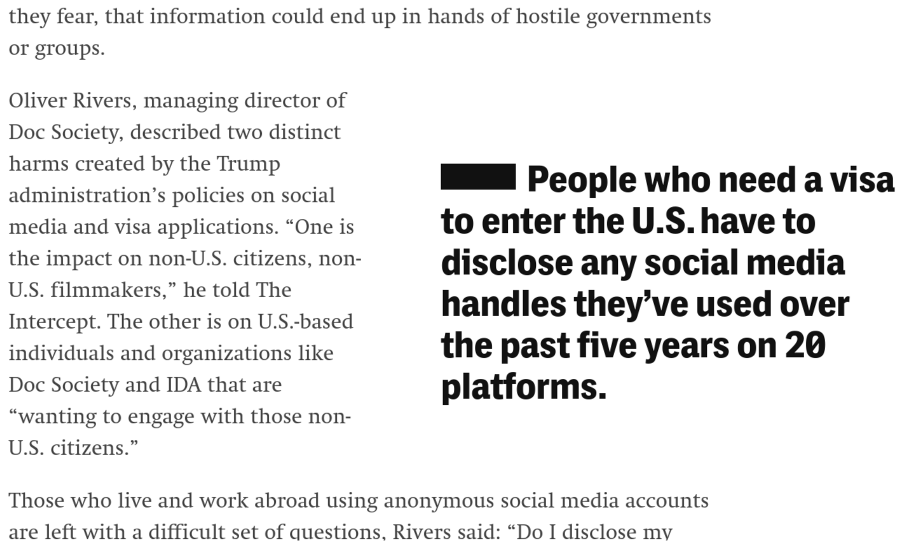
There also exist specific textual components of articles where the ability to spark situational interest is a defining characteristic:

Headlines Perhaps the most well-known component is the headline. Having a successful headline is crucial in attracting attention, and significant effort goes towards their optimization. This optimization increasingly occurs in an analytical and automated fashion [43]. One common way of measuring the success of an online headline is the clickthrough rate—the fraction of visitors who click on the headline after being exposed to it. The number of likes and shares on social media are also sometimes used [76]. A particular type of headline, known as **clickbait** may be employed to maximize the clickthrough rate of articles. Clickbait may be

[1] Muddying the distinct roles of coherence and situation interest in supporting internal motivation however, coherence may be a strong contributor to situational interest [32].

intentionally misleading to draw in visitors, even though reader expectations end up not being met, or they may employ a technique called “information gapping”, where key information is left out of the title, increasing curiosity [87, 43]. An overview of the types of techniques used by clickbait is provided by Yimin Chen et al. [13].

Pull quote Another textual component of engaging articles is the pull quote (PQ). PQs are graphical elements of articles with thought provoking spans of text pulled from an article by a writer or copy editor and presented on the page in a more salient manner [33], such as in Figure 2.1. Following the 15 year period between 1965 and 1980 where many newspapers experimented with their design (having previously been graphically similar) [107], some newspapers adopted a more modern design. Aspects of this newer design, preferred by readers, includes a more horizontal or modular layout, the six-column format, additional whitespace around heads, fewer stories, larger photographs, more colour, and more PQs [103, 109, 15]. PQs serve many purposes. They provide temptation (with unusual or intriguing phrases, they make strong entrypoints for a browsing reader), emphasis (by reinforcing particular aspects of the article), and improve overall visual balance and excitement [104, 49]. PQ frequency in reading material is also significantly related to information recall and student ratings of enjoyment, readability, and attractiveness [109, 110].



they fear, that information could end up in hands of hostile governments or groups.

Oliver Rivers, managing director of Doc Society, described two distinct harms created by the Trump administration’s policies on social media and visa applications. “One is the impact on non-U.S. citizens, non-U.S. filmmakers,” he told The Intercept. The other is on U.S.-based individuals and organizations like Doc Society and IDA that are “wanting to engage with those non-U.S. citizens.”

People who need a visa to enter the U.S. have to disclose any social media handles they’ve used over the past five years on 20 platforms.

Those who live and work abroad using anonymous social media accounts are left with a difficult set of questions, Rivers said: “Do I disclose my

Figure 2.1: An example of a PQ from `https://theintercept.com/2019/12/05/us-visa-social-media-screening/`.

In Chapter 4, we discuss machine learning approaches to identifying successful headlines or clickbait, as well as novel approaches for PQ selection. However, there are additional textual components of news articles with similar purposes A **strapline** is a second headline placed beneath the main headline used to highlight another point or amplify the main headline [95]. A **subhead** may refer to the same thing (i.e., a sub-headline), but can also refer to a one or two word headline placed in bold at the beginning of paragraphs [95]. A **kicker** may refer to several components, including the first few words of a caption to grab a reader’s attention [18]. The **lede** (also known as lead or lead paragraph) often refers to the first sentence or opening paragraph of an article [18, 68]. A good lede must not only grab the reader's attention (if the previously discussed mechanisms have not already done so, or build upon the interest they have generated), but supply the primary details of the article, i.e., the who, what, where, when, why, and how. By concisely providing the main details, a lede also functions as a summary of the article. The act of summarization, to varying degrees, shifts the priority away from situational interest and towards topic interest, as discussed in the next subsection.

2.3 Document Summarization

Document summarization is an important problem in NLP and appears, in one form or another, in all three articles forming the body of this thesis. At the highest level, summarization is the act of distilling something into a more compact form, while losing minimal information. This concept can be applied to many modalities and types of texts, including images [99, 112], videos [67], conversations [36], and computer event logs [40].

In the context of this thesis, summarization is important as a result of its ability to increase the personal interestingness of a text: of the information in a document, only a small fraction of it may be interesting to a given person. Summarization takes this diluted source of interestingness and distills it. Depending on the type of summarization employed, personal interestingness can be improved to varying degrees. The three main types of text summarization tasks of interest are described next.

Generic Summarization Generic document summarization (the most common type of summarization, often referred to simply as summarization) aims to capture and convey the main topics and key information discussed in a body of text in a generic way (the same for all readers) [81]. All types of summarization, including those described later, can be categorized as either extractive or abstractive. *Extractive* summarization aims to capture the most important information by extracting sentences. In contrast, *abstractive* summarization generates new text for the summary [82], and for this reason is often considered the more difficult type. While generic summarization is concerned with what is *important* or representative in a document, this overlaps with what is *interesting* when the text is relevant to the user. Approaches to generic summarization have roughly evolved from unsupervised (see Section 2.4 for descriptions of unsupervised and supervised learning) extractive heuristic-based methods [66, 70, 28, 83, 44], to supervised and often abstractive deep-learning approaches [80, 79, 78, 116]. In Chapter 3, as a side-effect of producing a novel coherence-modelling approach, we propose an approach to generic summarization which effectively works by identifying lede-like sentences.

Query-based Summarization For a given article, not every reader will be interested in the same parts. *Query-based summarization* (QS) aims to produce relevant summaries for personalized interests by generating a summary conditioned on a user-provided query [17, 82, 81]. For example, when applying QS to an article about cats with the query ”Why do cats meow?”, the summarizer will try to extract (or abstract) the information in the article relevant to why cats meow. Two flavors of query-based summaries are *informative* summaries, which provide the reader with the relevant answers, and *indicative* summaries, which provide the reader with a summary of what information is present, not necessarily the detailed information itself. Difficulties and approaches to QS are discussed further in Chapter 5.

Interactive Personalized Summarization While query-based summarization requires users to explicitly provide a query, *interactive personalized summarization* (IPS) considers the task where user preferences are learned from non-textual feedback [114]. This feedback can take many forms, both in terms of *how* users provide feedback (e.g., selecting, swiping, rating) and *what* they provide feedback on. For example, the APRIL IPS system [35] allows users to indicate their preference given pairs of candidate summaries. If the user repeatedly selects summaries mentioning a particular subset of topics, we can guess that a good personalized summary is one that focuses on those topics. In Chapter 5 we discuss IPS further, and propose a modified version of the IPS task, which we call HARE. This modified task aims to further reduce the effort and time required by users to produce a personalized summary by learning their preferences and summarizing the document *while* they read it.

These three types of summarization—generic, QS, and IPS—can be applied to single-document or multi-document settings. In this thesis, we are particularly interested in working with single documents at a time.

In the context of this thesis, relevance is the most important criterion for all types of summaries we consider. However, there are several other properties a good summary should possess:

- Coherence: As discussed in Section 2.2, coherence refers to how well the text is organized and structured. As extractive summaries work by selecting possibly non-contiguous sentences, it is possible that a resulting summary would not read very smoothly, with sentences feeling disconnected.

- Factual correctness: Also known as consistency, this property refers to the factual alignment between a summary and the source text [29]. A summary should only condense or abstract the information in a text, not fundamentally change it. For extractive summarizers, this is easy to achieve since no text is modified. For abstractive summarizers that generate new text, maintaining factual correctness has proven to be a challenge [63].

- Fluency: This property refers to the quality of individual sentences (grammar, spelling, capitalization, etc.) [29]. A good summarizer should of course produce text that is easy to read.

- Non-redundancy: Redundancy in a summary refers to the amount of information that is repeated [85]. This overlapping information can occur in cases of either exact textual overlap (which is more likely when performing multi-document summarization), or in overlap of concepts or details. In order to maximize the informativeness of a summary while minimizing its length, it should have minimal redundancy.

In the next section, we begin introducing concepts of machine learning, which can be used to automatically *learn* how to perform tasks such as coherence modelling, identifying interesting sentences, and summarization.

2.4 Fundamental Machine Learning Concepts

In this section, we provide a focused overview of machine learning, a field concerned with constructing computer programs that automatically learn from experience [75]. In particular, we will cover the concepts most relevant to understanding the integrated articles of this thesis, assuming a cursory knowledge of the field. First we will discuss two important learning paradigms: unsupervised and supervised learning. Second, we consider the central challenges of machine learning, and the various techniques that exist to overcome these challenges.

2.4.1 Learning Paradigms

Machine learning algorithms can be classified based on the types of information they learn from. The two types of information here refer to the feature space and the label space. When learning only from the feature space, algorithms are considered to be performing **unsupervised learning**. The aim of such algorithms is to uncover useful properties of the data. For example, clustering algorithms, such as K-Means, aim to identify naturally occurring structures within the data [113]. When clustering images of houses for example, the algorithm may end up grouping together images based on housing size or colour, time of day, and amount of greenery present. In Chapter 5, we apply clustering to embeddings (see Section 2.6) of sentences in a document to identify the general concepts present. For example, when clustering sentences in an article about cooking, we might identify core concepts related to the ingredients to prepare, health information, or the history of the recipe.

Supervised learning algorithms, in contrast, learn from labelled data. That is, a dataset that contains samples from the feature space (often called “inputs”) with corresponding labels (often called “outputs”). The goal of these algorithms is to find a function capable of predicting the label of a sample given a point in the feature space. In other words, learning an input–output mapping. Continuing the housing example, if we are wishing to construct a computer program capable of taking a picture of a house and predicting its price, then the label space would consist of possible prices (and each image constitutes a point in the feature space). If the possible prices lie along a continuum, it would be considered a *regression* problem, and if the labels instead come from a discrete set, it is a *classification* problem. Reviews of common machine learning models for supervised learning, including logistic regression, decision trees, k-nearest neighbors, and neural networks can be found in [75] by Tom Mitchell or [6] by Christopher Bishop. In Section 2.5 we will further discuss neural networks in particular. In our work, the clearest instances of supervised learning occur in Chapter 4, where we train classifiers to identify interesting sentences in a document.

Lying between unsupervised and supervised learning are **semi-supervised learning** and **self-supervised learning**. In semi-supervised learning, algorithms have access to both labelled and unlabelled data, with the unlabelled data generally being more voluminous [119]. The motivation for semi-supervised learning comes from the fact that unlabelled data is generally cheaper to obtain than labelled data, as labelling often requires human input. An example of an intuitive semi-supervised learning algorithm is label-propagation [120]. This algorithm is able to assign labels to unlabelled points in the feature space. The assigned label for a point is determined by the labels (either ground-truth or assigned) of any points nearby in the feature space. Self-supervised learning algorithms (which are often the same types of algorithms used by supervised learning) require only unlabelled data, and rely on surrogate tasks where the goal is to predict some properties of the feature space based on the remaining properties. Applications of self-supervised learning are commonly found in computer vision [51] and NLP [64], domains where there is often a whole lot of unlabelled data available, and the cost of labelling would be prohibitively high. From computer vision, one such task is *inpainting*, where some portion of an image is erased, and a model must learn to reconstruct the missing pixels given the remaining pixels. From NLP, one task is next-sentence prediction, whereby a model must learn to predict the next sentence given one or more preceding ones. This technique has been used to learn sentence embeddings useful for a large variety of tasks [22]. In this thesis, self-supervised learning appears in Chapter 3, where we propose a new self-supervision task: predicting the location of a sentence in a document given only semantic information.

2.4.2 Central Challenges of Machine Learning

An ever-present thought in the minds of machine learning practitioners is the **generalization** ability of fitted models. This is a measure of how well a model applies to unseen samples. In the case of supervised learning, generalization implies that the model is able to accurately predict the output for new inputs. For unsupervised learning, generalization implies that new samples are consistent with the identified patterns or structures. When the model over-simplifies the input-output relationship or underlying data distribution, it is said to **underfit**. On the other hand, when the model is overly sensitive to small changes or noise in the dataset, it is said to **overfit**. Figure 2.2 demonstrates simple examples of overfitting and underfitting in supervised and unsupervised scenarios.

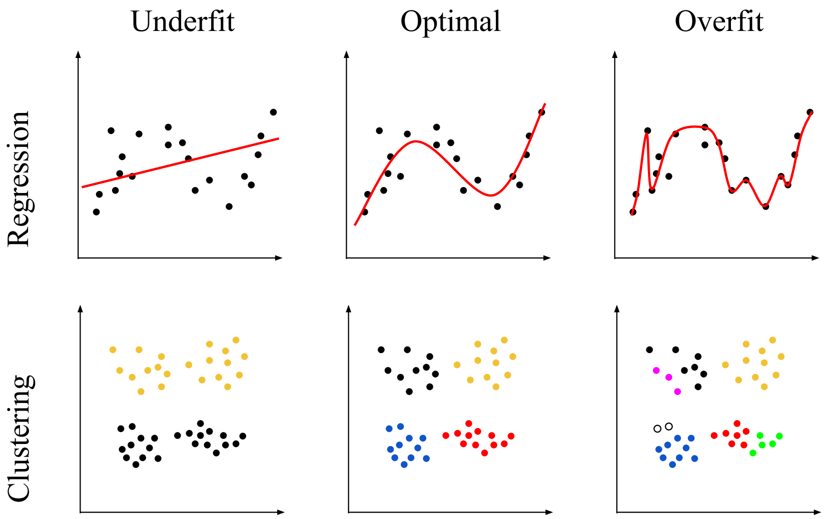


Figure 2.2: Examples of underfitting, optimal fitting, and overfitting in regression (supervised learning) and clustering (unsupervised learning). We consider a clustering model to overfit when it identifies structures in the data, such as additional "groups", that are likely the result of noise.

Traditionally, the problem of balancing underfitting and overfitting is understood to be a result of the **bias–variance trade-off**. This trade-off says that the more bias a model has (i.e., the more strongly the model assumes the data follows a predetermined pattern or family of patterns), the less variance it exhibits (i.e., sensitivity of the model predictions or learned structures to the particular samples in the dataset). By controlling the bias and variance of a model, via the *representational capacity* of the model, a "sweet spot" can be found between underfitting and overfitting [5], as reflected in the middle column of Figure 2.2. Representational capacity can be considered as the maximum complexity of patterns the model is able to fit.

Recently, the traditional understanding of the bias–variance trade-off and the connection between representational capacity and generalization ability has been under question. New machine learning models, large neural networks in particular, have been observed to follow a "double-descent" curve [5], whereby increasing model capacity arbitrarily beyond a certain threshold actually continues to *improve* generalization. For an in-depth treatment on this effect, see [5] by Mikhail Belkin et al.

There exist many techniques to tune model complexity, many of which are motivated by the traditional understanding of the bias–variance trade-off. With decision trees for example, we can increase the maximum complexity of the model through increasing the maximum tree depth. When training a logistic regression model, complexity can be tuned through setting the weight regularization strength, which penalizes the model having large weights. Weight regularization is discussed further in Section 2.5.2. Of course, by adding yet another hyperparameter such as maximum depth or regularization strength to a model, we could increase the overall variance if we are not careful. However, in supervised learning, we can select hyperparameters while avoiding overfitting through the use of **validation sets**. Rather than training many models with the various hyperparameter choices and evaluating them each on the testing data (which effectively tunes the hyperparameters to the test data), we can evaluate the models on an additional set of data specifically for hyperparameter validation. This allows our final evaluation of the best model on the test set to better reflect its true generalizability. Typically, the validation set is constructed with 20% of the training data, while the other 80% continues to be used only for training [38].

In the next section, we will provide background on neural networks, including hyperparameters used to mitigate generalization error.

2.5 Artificial Neural Networks

In this section, we provide an overview of artificial neural networks (which we refer to simply as *neural networks*). We start with the basic mathematics and core concepts, and then cover regularization techniques relevant to this thesis, and finally introduce several types of neural networks of particular interest.

2.5.1 Basics

To introduce neural networks, we can start from the *perceptron*, researched by Rosenblatt in the 1950s and 1960s [92, 93]. The perceptron is a primitive version of the artificial neuron which is the fundamental unit of a neural network.

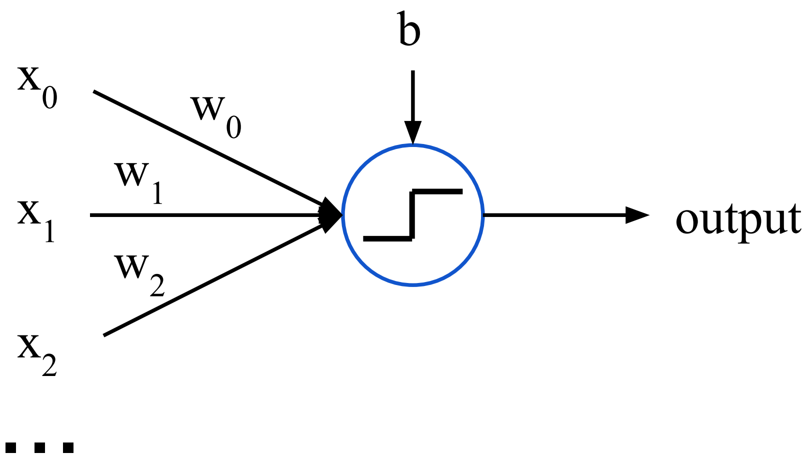


Figure 2.3: A perceptron, showing the input $(x_0, x_1, x_2, ...)$, the weights $(w_0, w_1, w_2, ...)$, and the bias, b.

The perceptron Loosely modelled after biological neurons, perceptrons are binary classifiers that can take in signals from several sources (analogous to dendrites in biological neurons), combine this information, and produce an output (similar to axons). This structure is shown in Figure 2.3, where the perceptron has 3 real-valued inputs: x_0, x_1, x_2. Each input, x_i, is also associated with a weight, w_i, which is the scaling factor of the given input (and can sometimes be interpreted as importance). To compute the output of the perceptron, we take the weighted sum of its inputs, plus a *bias*: $w_0x_0 + w_1x_1 + w_2x_2 + b$, which can be more compactly expressed as $\mathbf{w} \cdot \mathbf{x} + b$. If this weighted sum is larger than some threshold then the output is 1, otherwise it is 0. By modulating the bias, the sum can be biased towards either side of the threshold. This thresholding operation applied to the weighted sum is termed the *activation function*.

Modern artificial neurons Rather than using a simple thresholding operation, modern neurons use more expressive activation functions, such as:

- **sigmoid**: This function transforms values to lie between 0 and 1. This is useful for the output of a binary classifier: when the output is close to 0, it indicates the input is from class 0, and if it is close to 1, the input is from class 1. It is computed with the following equation:
$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$
- **tanh**: This function transforms values to lie between -1 and 1. It is computed with the following equation:
$$tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{2.2}$$
- **ReLU**: This piece-wise linear activation function simply returns 0 when the value is negative, and x where it is positive: $ReLU(x) = max(0, x)$. Despite its simplicity, it

brings several benefits over sigmoid and hyperbolic tangent activations. This includes increased biological plausibility, representation sparsity, and faster training of deep neural networks due to removing the *vanishing gradient problem* suffered by the previous activations [37].

- **softmax**: This function is frequently used as the output activation function when performing multi-class classification, as it returns a vector whose values sum to 1, allowing them to be interpreted as class probabilities. For each value, x_i of the vector, x, it is computed with the following equation:

$$softmax(x_i) = \frac{exp(x_i)}{\sum_j exp(x_j)} \tag{2.3}$$

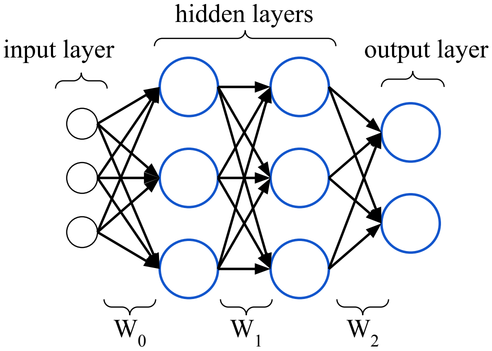


Figure 2.4: Multi-layer neural network (biases not shown).

Layers of neurons Modern neural networks consist of many neurons, often arranged in *layers*. Neurons used to input data to the network are are part of the *input layer*. Those neurons which produce the final output are in the *output layer*. Those neurons (if any) which lie between the input and output are in *hidden layers*. The number of neurons in each layer is referred to as the width, and the total number of layer is the depth. A small example of a multi-layer neural network is shown in Figure 2.4. The matrix W_i corresponding to the weights of the neurons in layer i (one row per neuron). The full set of weights which parameterize a neural network is often represented by θ, and the function which represents the network is h_θ. If we use the activation function f and ignore the biases for simplicity, the output of the neural network in the figure for a given input X is computed with:

$$h_\theta(X) = f(W_2 f(W_1 f(W_0 X))) \tag{2.4}$$

Optimization For neural networks to be useful at solving a classification or regression task, they should form a close approximation of the true function mapping the input features, X, to the labels, y. To measure how far away the network is from providing a good approximation of this function, we can use *loss functions*. Given a neural network, h_θ, loss functions take the predictions of a network, $y' = h_\theta(X)$, and compare these predictions to the true labels. Often used for regression tasks, the **mean squared error** loss simply computes the square of the difference between the true and predicted values, averaged over the n samples:

$$MSE(y, y') = \frac{1}{n} \sum_{i=1}^{n} (y_i - y_i')^2 \tag{2.5}$$

For classification tasks, **cross entropy** loss is often used, which provides a measure of the distance between two probability distributions. If we consider y_i' to be the predicted discrete probability distribution over all classes, and y_i to be the distribution where all mass is placed on the single correct class (and all other values are 0), then the loss is computed with:

$$CE(y, y') = -\frac{1}{n} \sum_{i=1}^{n} y_i \cdot \log(y_i') \tag{2.6}$$

In the particular case of binary classification, where each y_i is 1 or 0, then this is often written:

$$BinaryCE(y, y') = -\frac{1}{n} \sum_{i=1}^{n} (y_i \log(y_i') + (1 - y_i) \log(1 - y_i')) \tag{2.7}$$

In order to augment the weights of a neural network, θ, such that the loss is minimized, we can use a variety of training methods called *optimizers*. The most popular class of optimizers combine *stochastic gradient descent* (SGD) and *back-propagation*. SGD is based on the gradient descent method of optimization, where the idea is to compute the gradient of the loss with respect to the model parameters, and step in the direction of steepest descent:

$$\theta \leftarrow \theta - \epsilon \nabla_{\theta} J(\theta) \tag{2.8}$$

Here, $J(\theta)$ refers to our loss to minimize, as a function of θ. The *learning rate*, ϵ, controls the step size. Rather than compute the gradient based on the entire dataset for each step (which can be very slow or computationally infeasible), SGD uses an approximation of the the true gradient by computing it for each sample. As this often produces a much more noisy gradient, it is common to use *minibatch* SGD, which averages the gradient over a batch of samples for each step. The number of samples in each batch is termed the *batch size*. Many modifications to SGD have been proposed to help train neural networks, which incorporate concepts such as momentum and adaptive learning rates [88, 54]. Back-propagation provides the particular method for computing the gradient with respect to each model parameter, and works through recursively applying the chain rule of calculus [94].

In addition to SGD, other types of optimization include Hebbian learning [45, 1], reinforcement learning [105, 100], and evolution [115, 71, 34].

2.5.2 Regularization

Neural networks, especially deep neural networks, are often *overparamaterized*, which means that the number of weights which require tuning is larger than the number of training samples available. This means that for a given dataset, (X, y), there may be an infinite number of choices for the weights which perfectly capture the mapping. In order to discover solutions that are more likely to generalize to new data, various regularization techniques can be applied.

Weight regularization One of the oldest techniques is weight regularization, in the form of L_1 or L_2. When L_1 regularization is used, the loss function not only encourages accurately predicting y, but also minimizing $|\theta|$. This induces *sparsity* in the weights, intending to capture our intuition that not all features are useful for a given problem (e.g., when predicting housing prices, we can probably ignore the radius of the doorknobs). L_2 similarly constrains the values of the weights by trying to minimize $\theta^T\theta$, which decreases the ability of the model to easily overfit to noise in the data.

Dropout This is a common regularization technique for neural networks which is intended to prevent the "co-adaptation of features" [47, 102]. This is achieved by randomly disabling (or "dropping out") a fraction of neurons and their connections in each layer of a network during training. This encourages the learning of robust and diverse feature detectors, as the network may not be able to assume the constant presence of all neurons responsible for detecting any given feature. Dropout is only applied during the training stage and is disabled during testing. With standard dropout, if we keep weights or neurons with a probability of p, then we need to re-scale weights by by a factor of p during testing to account for the fact that each neuron now receives an increased number of inputs. Alternatively, there is *inverted* dropout, where weights are re-scaled during training by a factor of $1/p$.

Early-Stopping We often observe that as a neural network is trained, at first both the training loss and validation loss decrease. As training continues however, eventually the validation loss begins to increase, while the training loss continues to decrease, signalling that the model is overfitting to particular aspects or noise in the training data that is not present in the validation data. To stop training at the point where validation loss is minimized, we can perform early-stopping. This is achieved by monitoring the validation loss during training, and stopping training once the validation loss stops decreasing [38].

2.5.3 Important Types of Neural Networks

Beyond the basic type of neural network architecture, the *feedforward fully-connected* network (such as in Figure 2.4), there is a veritable zoo of additional established types of networks[2]. Each type of network is constructed to incorporate or avoid various inductive biases, making it easier to achieve high accuracy given different amounts of data (with less training data generally calling for stronger inductive biases). Perhaps the most well-known example of the success of useful inductive biases in neural networks is from *convolutional neural networks* (CNNs) [60], commonly used for computer vision tasks. If a human is to specify whether an image contains a picture of a cat, we intuitively know both that the pixels representing a cat will tend to have a particular spatial organization, and that the answer does not depend on the precise location of the cat in the image. CNNs, which rely heavily upon feature maps and the shift invariant convolution operation, make use of these observations [61, 118].

While the work in this thesis does not use CNNs, there are two other important types with their own inductive biases that directly appear in this work:

[2] For a large visual collection, see `https://www.asimovinstitute.org/neural-network-zoo/`.

Neural mixture-of-experts While this type of network will also be described in Chapter 4, briefly, it assumes that the problem to be solved is best represented as multiple distinct but related problems. In a mixture-of-experts model, an "expert" is trained for each sub-problem, and a *gating function* decides, for a given input, which expert (or weighting over the experts) to rely on [50]. For example, if we want a single large neural network to classify images of animals into their species, it may be beneficial for one expert (each expert constitutes a sub-network) to specialize in classifying mammals, one specialize in birds, one in insects, etc. The strength of a *neural* mixture-of-experts is that the areas of specialization for each expert are automatically decided. In Chapter 4, we find that using a neural mixture-of-experts outperforms simpler architectures in identifying whether sentences in news articles are pull quotes or not.

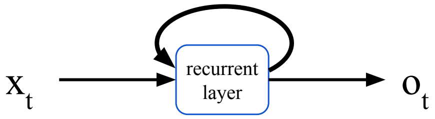


Figure 2.5: A simplified diagram of a single-layer recurrent network. At each time step, it receives the next element of the input sequence as well as the layer output from the previous time step.

Recurrent neural networks While CNNs assume spatial structure in the input, recurrent neural networks (RNNs) assume a sequential or temporal aspect [94]. Additionally, while feedforward densely-connected networks only accept a fixed-size input and produce a fixed-size output, RNNs can accept arbitrary-length input and produce an arbitrary-length output[3]. This is especially useful for problems in signal processing or NLP, where sequences of words or characters are abundant. If we wish to predict the part-of-speech tags for each word in a sentence for example, RNNs are especially effective, as their inductive bias allows them to easily capture certain hierarchical syntax structures [106].

The fundamental idea behind RNNs is that, instead of processing each element, x_t, of an input sequence, $x = (x_0, x_1, ...)$, independently, we want to condition the processing based on all previously seen elements. This is enabled through recurrent connections, as shown in Figure 2.5, where the output of a layer at time step $t - 1$ becomes part of the input at time t. Various extensions of this concept have been proposed to allow RNNs to more efficiently learn patterns over longer gaps of time, with the most popular being Long Short-Term Memory networks (LSTMs) [48]. For a full treatment of RNNs and LSTMs, see [39] by Alex Graves.

In this thesis, we use RNNs in Chapter 3 to learn to map a sequence of words to a single output vector.

[3] To improve training efficiency however, it is common to specify a maximum sequence length, and trim or pad sequences as necessary.

2.6 Text Embeddings

In the previous sections, we have introduced the areas of NLP that we are interested in working in, as well as machine learning concepts which will eventually be used to solve problems in those areas. In this section we will cover text embeddings, which form the bridge between the problem domain of NLP and machine learning algorithms. Text embeddings achieve this by converting unstructured text into vector representations that can be used by learning algorithms. We will introduce three general types of embeddings that appear in this thesis: handcrafted features, n-gram representations, and distributed embeddings.

2.6.1 Handcrafted Features

The first type of embedding makes use of manually defined features. Simple examples of handcrafted features that can be concatenated into a word embedding include word length, frequency, part-of-speech tag, sentiment, and number of syllables. For embedding sentences, we can simply average across the embeddings of individual words, or we can also use sentence-specific handcrafted features such as total sentence length, reading difficulty, argumentative purpose, sentiment (the sentiment of a whole sentence is often a complex function of the individual words), or location in the document. When we have a strong intuition about what features are likely to be important when solving a problem, handcrafted features may work well. However, for more difficult problems, the embedding techniques discussed next are more popular.

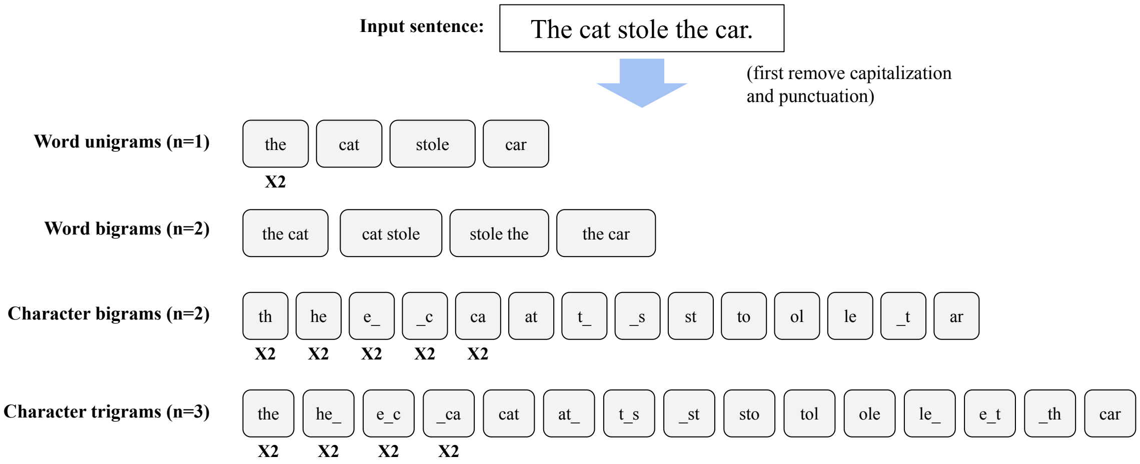


Figure 2.6: Converting a sentence into various word and character level n-grams.

2.6.2 n-Gram Representations

An *n-gram* is a contiguous sequence of n tokens from a longer sequence. When embedding sentences or documents, these tokens are often characters or words. Figure 2.6 demonstrates how to decompose a sentence into a series of n-grams for both characters and words, for $n \in [1, 2, 3]$. To create the actual embedding for a piece of text, we simply count the occurrences of each n-gram and place these values into a vector, which removes positional information of the n-grams. When $n = 1$ and our tokens are words, this technique is equivalent to *bag-of-words* (BOW).

Since the number of possible n-grams grows exponentially with n and token vocabularies may be very large, we can limit our n-gram vocabulary (thus the embedding dimension) by only considering occurrences of the N most frequent n-grams. For example, when using word bigrams ($n = 2$), we can avoid requiring an embedding dimension corresponding to occurrences of the phrase "predator dishwasher", since it is unlikely to occur at all. When working with words, it is also common to remove *stopwords*, which are frequent words that carry little information (such as "the", "and", "a", etc.).

Despite its simplicity, n-gram representations are still commonly used, and often make for a strong baseline on text classification problems [52, 42]. Our own work in Chapter 4 supports this, demonstrating that when combined with a simple classifier, n-grams help achieve near-best performance on PQ selection.

2.6.3 Distributed Embeddings

With handcrafted features, we arrive at the problem of having to decide ourselves what features will be important, and perform the often complex task of determining how to compute those features. If we do not already know how to solve the problem, it is unlikely that we can be confident on what features will work best. With n-gram embeddings, we have the problem of sparsity (most embedding dimensions will be 0), which has the side-effect that, for example, if two sentences are semantically similar but use different exact words, their embeddings will be far apart. For example, the two phrases have roughly the same meaning with minimal word overlap: 1) "The black cat slept.", 2) "The charcoal feline snoozed." Distributed text embeddings try to solve both of these problems through automatically discovering useful embedding dimensions and ensuring that distance in the embedding space roughly corresponds to semantic distance.

For an intuitive introduction to distributed embeddings, we can consider the influential *word2vec* approach proposed by Mikolov et al. [73]. This word embedding technique is driven by the intuition that words with similar meaning will have similar contexts (surrounding words). This intuition is exploited by training a model with the *Skip-gram* architecture which has a single hidden layer with a linear activation, as shown in Figure 2.7. This model is trained to take a *one-hot encoding* of a word from a text as input, project it to a lower-dimensional embedding with the hidden layer, and use the embedding to predict the one-hot encodings of the words both to the left and right of it. For a vocabulary of size N, the one-hot encoding of a word is simply the N-dimensional vector where only the dimension corresponding to the index of the word in the vocabulary is 1 and all others are zero.

Once the Skip-gram model is trained (which is done with a suite of tricks for improved efficiency [73]), the word2vec embedding of a word is obtained by recording the output of the hidden layer. If two words have similar contexts, then their embeddings must be similar. Additionally, by having a high-dimensional word embedding space, many types of word properties can be encoded. These embeddings can capture gender for example: if $vec(king)$ represents the distributed embedding of "king", and $vec(man)$, is the embedding for "man", then these embeddings match our intuition in that $vec(king) - vec(man) + vec(woman) \approx vec(queen)$ [74].

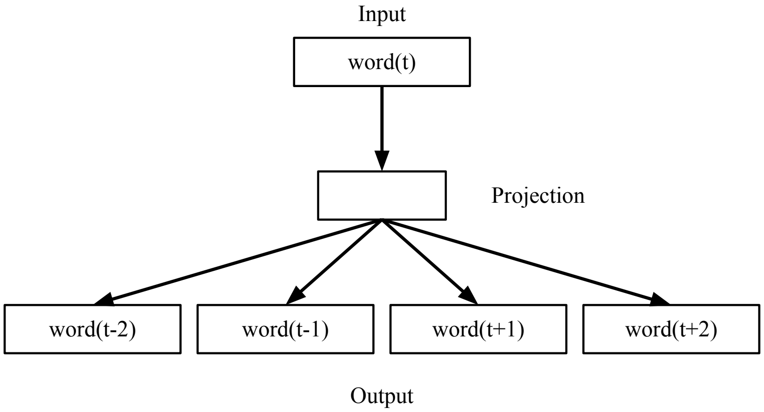


Figure 2.7: Skip-gram architecture with a context size of 2 (two words before and after an input word). By training a neural network to predict the words surrounding an input word, we can learn useful distributed word representations.

Additional examples of word relationships captured by word2vec embeddings are provided by Mikolov et al. [73]. In this thesis, we made use of fastText word embeddings [52] in Chapter 3, which can be seen as an improvement upon word2vec. A primary difference is that instead of using the Skip-gram architecture, it uses the Continuous Bag-of-Words (CBOW) architecture, where we input the surrounding context of a word, and train the model to predict the center word [72].

In addition to distributed word embeddings, there are also distributed sentence [55, 16, 22, 90] and document embeddings [59]. These techniques tend to rely on the same general concept as word2vec: given part of the information from some context window, predict the remainder. The skip-thought model for example is trained to take a sentence and predict the embeddings of those to the left and right [55]. The BERT (Bidirectional Encoder Representations from Transformers) model makes use of two tasks to learn sentence embeddings: 1) randomly masking some of the input words and trying to predict their vocabulary id based on the remaining unmasked words, and 2) "next sentence prediction" (determining whether one of the input sentences comes after the other input sentence) [22]. In this thesis, we make use of the Sentence-BERT (SBERT) embedding model, a modification of BERT which allows for efficiently extracting more semantically meaningful sentence embeddings [90]. SBERT achieved state-of-the-art performance on several semantic textual similarity tasks when it was released in 2019. SBERT is used in Chapter 4 and Chapter 5.

Chapter 3

Paper 1

A version of this paper, titled “Learning Sentence Embeddings for Coherence Modelling and Beyond”, was submitted to and published by the International Conference on Recent Advances in Natural Language Processing (RANLP) 2019. It was accepted for poster presentation and publication in the conference proceedings [7].

This paper presents a new technique for learning sentence embeddings which prove useful for multiple coherence-related tasks. They can be used to visually estimate the coherence of an article (and thus, an aspect of its readability), and identify *where* the incoherent sentence transitions are likely to be. It is worth noting that this paper considers coherence at the scale of sentences. If a sentence lacks coherence at the level of individual words, our technique may not recognize it. Our technique is instead able to identify when a sentence is located in an unusual position in the article.

Learning Sentence Embeddings for Coherence Modelling and Beyond

Abstract

We present a novel and effective technique for performing text coherence tasks while facilitating deeper insights into the data. Despite obtaining ever-increasing task performance, modern deep-learning approaches to NLP tasks often only provide users with the final network decision and no additional understanding of the data. In this work, we show that a new type of sentence embedding learned through self-supervision can be applied effectively to text coherence tasks while serving as a window through which deeper understanding of the data can be obtained. To produce these sentence embeddings, we train a recurrent neural network to take individual sentences and predict their location in a document in the form of a distribution over locations. We demonstrate that these embeddings, combined with simple visual heuristics, can be used to achieve performance competitive with state-of-the-art on multiple text coherence tasks, outperforming more complex and specialized approaches. Additionally, we demonstrate that these embeddings can provide insights useful to writers for improving writing quality and informing document structuring, and assisting readers in summarizing and locating information.

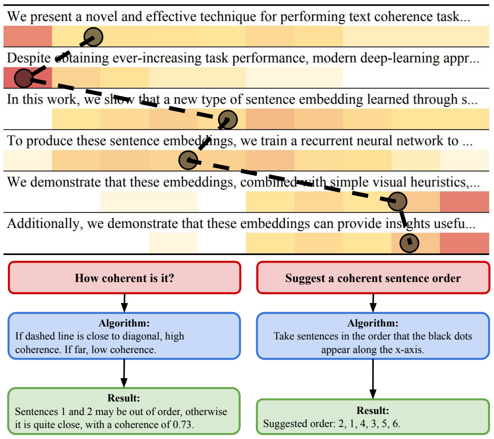

We present a novel and effective technique for performing text coherence task...
Despite obtaining ever-increasing task performance, modern deep-learning appr...
In this work, we show that a new type of sentence embedding learned through s...
To produce these sentence embeddings, we train a recurrent neural network to ...
We demonstrate that these embeddings, combined with simple visual heuristics,...
Additionally, we demonstrate that these embeddings can provide insights usefu...
How coherent is it?
Algorithm:
If dashed line is close to diagonal, high coherence. If far, low coherence.
Result:
Sentences 1 and 2 may be out of order, otherwise it is quite close, with a coherence of 0.73.
Suggest a coherent sentence order
Algorithm:
Take sentences in the order that the black dots appear along the x-axis.
Result:
Suggested order: 2, 1, 4, 3, 5, 6.

Figure 1: This paper abstract is analyzed by our sentence position model trained on academic abstracts. The sentence encodings (predicted position distributions) are shown below each sentence, where white is low probability and red is high. Position quantiles are ordered from left to right. The first sentence, for example, is typical of the first sentence of abstracts as reflected in the high first-quantile value. For two text coherence tasks, we show the how the sentence encodings can easily be used to solve them. The black dots indicate the weighted average predicted position for each sentence.

1 Introduction

A goal of much of NLP research is to create tools that not only assist in completing tasks, but help gain insights into the text being analyzed. This is especially true of text coherence tasks, as users are likely to wonder where efforts should be focused to improve writing or understand how text should be reorganized for improved coherence. By improving coherence, a text becomes easier to read and understand (Lapata and Barzilay, 2005), and in this work we particularly focus on measuring coherence in terms of sentence ordering.

Many recent approaches to NLP tasks make use of end-to-end neural approaches which exhibit ever-increasing performance, but provide little value to end-users beyond a classification or regression value (Gong et al., 2016; Logeswaran et al., 2018; Cui et al., 2018). This leaves open the question of whether we can achieve good performance on NLP tasks while simultaneously providing users with easily obtainable insights into the data. This is precisely what the work in this paper aims to do in the context of coherence analysis, by providing a tool with which users can quickly and visually gain insight into structural information about a text. To accomplish this, we rely on the surprising importance of sentence location in many areas of natural language processing. If a sentence does not appear to belong where it is located, it decreases the coherence and readability of the text (Lapata and Barzilay, 2005). If a sentence is located at the beginning of a document or news article, it is very likely to be a part of a high quality extractive summary (See et al., 2017). The location of a sentence in a scientific abstract is also an informative indicator of its rhetorical purpose (Teufel et al., 1999). It thus follows that the knowledge of where a sentence should be located in a text is valuable.

Tasks requiring knowledge of sentence position – both relative to neighboring sentences and globally – appear in text coherence modelling, with two important tasks being order discrimination (is a sequence of sentences in the correct order?) and sentence ordering (re-order a set of unordered sentences). Traditional methods in this area make use of manual feature engineering and established theory behind coherence (Lapata and Barzilay, 2005; Barzilay and Lapata, 2008; Grosz et al., 1995). Modern deep-learning based approaches to these tasks tend to revolve around taking raw words and directly predicting local (Li and Hovy, 2014; Chen et al., 2016) or global (Cui et al., 2017; Li and Jurafsky, 2017) coherence scores or directly output a coherent sentence ordering (Gong et al., 2016; Logeswaran et al., 2018; Cui et al., 2018). While new deep-learning based approaches in text coherence continue to achieve ever-increasing performance, their value in real-world applications is undermined by the lack of actionable insights made available to users.

In this paper, we introduce a self-supervised approach for learning sentence embeddings which can be used effectively for text coherence tasks (Section 3) while also facilitating deeper understanding of the data (Section 4). Figure 1 provides a taste of this, displaying the sentence embeddings for the abstract of this paper. The self-supervision task we employ is that of predicting the location of a sentence in a document given only the raw text. By training a neural network on this task, it is forced to learn how the location of a sentence in a structured text is related to its syntax and semantics. As a neural model, we use a bi-directional recurrent neural network, and train it to take sentences and predict a discrete distribution over possible locations in the source text. We demonstrate the effectiveness of predicted position distributions as an accurate way to assess document coherence by performing order discrimination and sentence reordering of scientific abstracts. We also demonstrate a few types of insights that these embeddings make available to users that the predicted location of a sentence in a news article can be used to formulate an effective heuristic for extractive document summarization – outperforming existing heuristic methods.

The primary contributions of this work are thus:

1. We propose a novel self-supervised approach to learn sentence embeddings which works by learning to map sentences to a distribution over positions in a document (Section 2.2).

2. We describe how these sentence embeddings can be applied to established coherence tasks using simple algorithms amenable to visual approximation (Section 2.3).

3. We demonstrate that these embeddings are competitive at solving text coherence tasks (Section 3) while quickly providing access to further insights into texts (Section 4).

2 Predicted Position Distributions

2.1 Overview

By training a machine learning model to predict the location of a sentence in a body of text (conditioned upon features not trivially indicative of position), we obtain a sentence position model such that sentences predicted to be at a particular location possess properties typical of sentences found at that position[1]. For example, if a sentence is predicted to be at the beginning of a news article, it should resemble an introductory sentence.

In the remainder of this section we describe our neural sentence position model and then discuss how it can be applied to text coherence tasks.

[1] If we instead learned to *rank* sentences, we would lose this ability to learn about normative properties of sentences, as a first-rank sentence does not necessarily mean at-the-beginning.

2.2 Neural Position Model

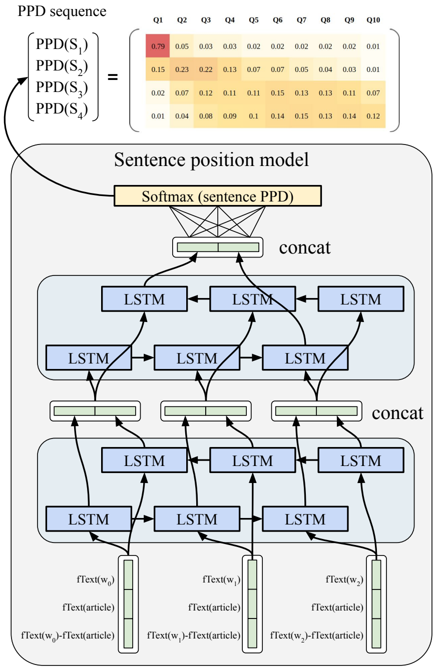


Figure 2: Illustration of the sentence position model, consisting of stacked BiLSTMs. Sentences from a text are individually fed into the model to produce a PPD sequence. In this diagram we see a word sequence of length three fed into the model, which will output a single row in the PPD sequence.

The purpose of the position model is to produce sentence embeddings by predicting the position in a text of a given sentence. Training this model requires no manual labeling, needing only samples of text from the target domain. By discovering patterns in this data, the model produces sentence embeddings suitable for a variety of coherence-related NLP tasks.

2.2.1 Model Architecture

To implement the position model, we use stacked bi-directional LSTMs (Schuster and Paliwal, 1997) followed by a softmax output layer. Instead of predicting a single continuous value for the position of a sentence as the fraction of the way through a document, we frame sentence position prediction as a classification problem.

Framing the position prediction task as classification was initially motivated by the poor performance of regression models; since the task of position prediction is quite difficult, we observed that regression models would consistently make predictions very close to 0.5 (middle of the document), thus not providing much useful information. To convert the task to a classification problem, we aim to determine what quantile of the document a sentence resides in. Notationally, we will refer to the number of quantiles as Q. We can interpret the class probabilities behind a prediction as a discrete distribution over positions for a sentence, providing us with a predicted position distribution (PPD). When $Q = 2$ for example, we are predicting whether a sentence is in the first or last half of a document. When $Q = 4$, we are predicting which quarter of the document it is in. In Figure 2 is a visualization of the neural architecture which produces PPDs of $Q = 10$.

2.2.2 Features Used

The sentence position model receives an input sentence as a sequence of word encodings and outputs a single vector of dimension Q. Sentences are fed into the BiLSTM one at a time as a sequence of word encodings, where the encoding for each word consists of the concatenation of: (1) a pretrained word embedding, (2) the average of the pretrained word embedding for the entire document (which is constant for all words in a document), and (3) the difference of the first two components (although this information is learnable given the first two components, we found during early experimentation that it confers a small performance improvement). In addition to our own observations, the document-wide average component was also shown in (Logeswaran et al., 2018) to improve performance at sentence ordering, a task similar to sentence location prediction. For the pretrained word embeddings, we use 300 dimensional fastText embeddings[2], shown to have excellent cross-task performance (Joulin et al., 2016). In Figure 2, the notation $ftxt(token)$ represents converting a textual token (word or document) to its fastText embedding. The embedding for a document is the average of the embeddings for all words in it.

The features composing the sentence embeddings fed into the position model must be chosen carefully so that the order of the sentences does not directly affect the embeddings (i.e. the sentence embeddings should be the same whether the

[2] Available online at https://fasttext.cc/docs/en/english-vectors.html. We used the wiki-news-300d-1M vectors.

sentence ordering is permuted or not). This is because we want the predicted sentence positions to be independent of the true sentence position, and not every sentence embedding technique provides this. As a simple example, if we include the true location of a sentence in a text as a feature when training the position model, then instead of learning the connection between sentence meaning and position, the mapping would trivially exploit the known sentence position to perfectly predict the sentence quantile position. This would not allow us to observe where the sentence *seems* it should be located.

2.3 Application to Coherence Tasks

For the tasks of both sentence ordering and calculating coherence, PPDs can be combined with simple visually intuitive heuristics, as demonstrated in Figure 3.

2.3.1 Sentence Ordering

To induce a new ordering on a sequence of sentences, S, we simply sort the sentence by their weighted average predicted quantile, $\hat{\mathcal{Q}}(s \in S)$, defined by:

$$\hat{\mathcal{Q}}(s) = \sum_{i=1}^{Q} i \times PPD(s)_i, \qquad (1)$$

where $PPD(s)$ is the Q-dimensional predicted position distribution/sentence embedding for the sentence s.

2.3.2 Calculating coherence

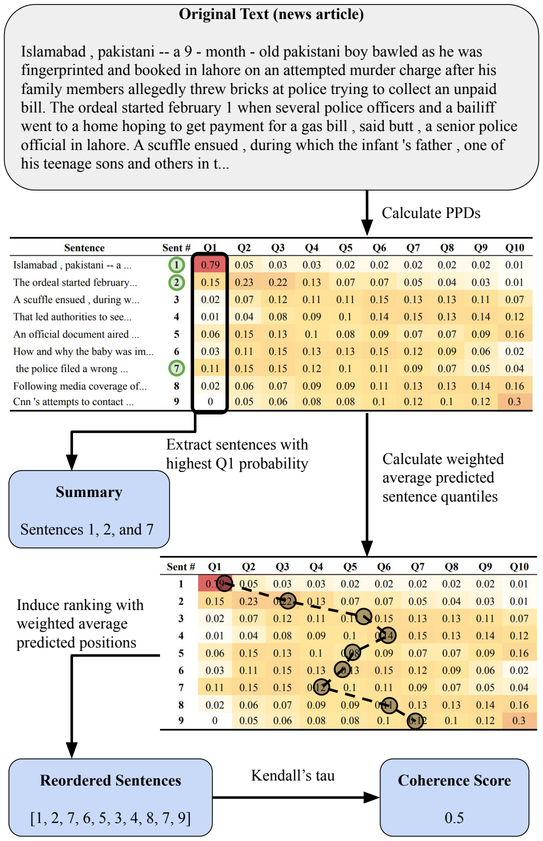
To calculate the coherence of a text, we employ the following simple algorithm on top of the PPDs: use the Kendall's tau coefficient between the sentence ordering induced by the weighted average predicted sentence positions and the true sentence positions:

$$coh = \tau((\hat{\mathcal{Q}}(s),\ \text{for } s = S_1, ..., S_{|S|}), (1, ..., |S|)). \qquad (2)$$

3 Experiments

In this section, we evaluate our PPD-based approaches on two coherence tasks and demonstrate that only minimal performance is given up by our approach to providing more insightful sentence embeddings.

Order discrimination setup. For order discrimination, we use the Accidents and Earthquakes datasets from (Barzilay and Lapata, 2008) which consists of aviation accident reports and news articles related to earthquakes respectively. The task is to determine which of a permuted ordering of the sentences and the original ordering is the most coherent (in the original order), for twenty such permutations. Since these datasets only contain training and testing partitions, we follow (Li and Hovy, 2014) and perform 10-fold cross-validation for hyperparameter tuning. Performance is measured with the accuracy with which the permuted sentences are identified. For example, the Entity Grid baseline in Table 2 gets 90.4% accuracy because given a shuffled report and original report, it correctly classifies them 90.4% of the time.



Original Text (news article)

Islamabad , pakistani -- a 9 - month - old pakistani boy bawled as he was fingerprinted and booked in lahore on an attempted murder charge after his family members allegedly threw bricks at police trying to collect an unpaid bill. The ordeal started february 1 when several police officers and a bailiff went to a home hoping to get payment for a gas bill , said butt , a senior police official in lahore. A scuffle ensued , during which the infant 's father , one of his teenage sons and others in t...

Sentence	Sent #	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
Islamabad , pakistani -- a ...	1	0.79	0.05	0.03	0.03	0.02	0.02	0.02	0.02	0.02	0.01
The ordeal started february...	2	0.15	0.23	0.22	0.13	0.07	0.07	0.05	0.04	0.03	0.01
A scuffle ensued , during w...	3	0.02	0.07	0.12	0.11	0.11	0.15	0.13	0.13	0.11	0.07
That led authorities to see...	4	0.01	0.04	0.08	0.09	0.1	0.14	0.15	0.13	0.14	0.12
An official document aired ...	5	0.06	0.15	0.13	0.1	0.08	0.09	0.07	0.07	0.09	0.16
How and why the baby was im...	6	0.03	0.11	0.15	0.13	0.13	0.15	0.12	0.09	0.06	0.02
the police filed a wrong ...	7	0.11	0.15	0.15	0.12	0.1	0.11	0.09	0.07	0.05	0.04
Following media coverage of...	8	0.02	0.06	0.07	0.09	0.09	0.11	0.13	0.13	0.14	0.16
Cnn 's attempts to contact ...	9	0	0.05	0.06	0.08	0.08	0.1	0.12	0.1	0.12	0.3

Extract sentences with highest Q1 probability
Calculate weighted average predicted sentence quantiles

Summary

Sentences 1, 2, and 7

Sent #	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
1	0.79	0.05	0.03	0.03	0.02	0.02	0.02	0.02	0.02	0.01
2	0.15	0.23	0.22	0.13	0.07	0.07	0.05	0.04	0.03	0.01
3	0.02	0.07	0.12	0.11	0.11	0.15	0.13	0.13	0.11	0.07
4	0.01	0.04	0.08	0.09	0.1	0.14	0.15	0.13	0.14	0.12
5	0.06	0.15	0.13	0.1	0.08	0.09	0.07	0.07	0.09	0.16
6	0.03	0.11	0.15	0.13	0.13	0.15	0.12	0.09	0.06	0.02
7	0.11	0.15	0.15	0.12	0.1	0.11	0.09	0.07	0.05	0.04
8	0.02	0.06	0.07	0.09	0.09	0.11	0.13	0.13	0.14	0.16
9	0	0.05	0.06	0.08	0.08	0.1	0.12	0.1	0.12	0.3

Reordered Sentences

[1, 2, 7, 6, 5, 3, 4, 8, 7, 9]

Coherence Score

0.5

Figure 3: A visualization of our NLP algorithms utilizing PPDs applied to a news article. To reorder sentences, we calculate average weighted positions (identified with black circles) to induce an ordering. Coherence is calculated with the Kendall's rank correlation coefficient between the true and induced ranking. We also show how PPDs can be used to perform summarization, as we will explore further in Section 4.

Task	Dataset	Q	Epochs	Layer dropouts	Layer widths
Order Disrcim.	Accident	5	10	(0.4, 0.2)	(256, 256)
	Earthquake	10	5	(0.4, 0.2)	(256, 64)
Reordering	NeurIPS	15	20	(0.5, 0.25)	(256, 256)

Table 1: The neural sentence position model hyperparameters used in our coherence experiments. The following settings are used across all tasks: batch size of 32, sentence trimming/padding to a length of 25 words, the vocabulary is set to the 1000 most frequent words in the associated training set. The Adamax optimizer is used (Kingma and Ba, 2014) with default parameters supplied by Keras (Chollet et al., 2015).

Sentence ordering setup. For sentence ordering, we use past NeurIPS abstracts to compare with previous works. While our validation and test partitions are nearly identical to those from (Logeswaran et al., 2018), we use a publicly available dataset[3] which is missing the years 2005, 2006, and 2007 from the training set ((Logeswaran et al., 2018) collected data from 2005 - 2013). Abstracts from 2014 are used for validation, and 2015 is used for testing. To measure performance, we report both reordered sentence position accuracy as well as Kendall's rank correlation coefficient. For example, the Random baseline correctly predicts the index of sentences 15.6% of the time, but there is no correlation between the predicted ordering and true ordering, so $\tau = 0$.

Training and tuning. Hyperparameter tuning for both tasks is done with a random search, choosing the hyperparameter set with the best validation score averaged across the 10 folds for order discrimination dataset and for three trials for the sentence reordering task. The final hyperparameters chosen are in Table 1.

Baselines. We compare our results against a random baseline, the traditional Entity Grid approach from (Barzilay and Lapata, 2008), Window network (Li and Hovy, 2014), LSTM+PtrNet (Gong et al., 2016), RNN Decoder and Varient-LSTM+PtrNet from (Logeswaran et al., 2018), and the most recent state-of-the art ATTOrderNet (Cui et al., 2018).

Results. Results for both coherence tasks are collected in Table 2. For the order discrimination task, we find that on both datasets, our PPD-based approach only slightly underperforms ATTOrderNet (Cui et al., 2018), with performance similar to the LSTM+PtrNet approaches (Gong et al., 2016; Logeswaran et al., 2018). On the more difficult sentence reordering task, our approach exhibits performance closer to the state-of-the-art, achieving the same ranking correlation and only slightly lower positional accuracy. Given that the publicly available training set for the reordering task is slightly smaller than that used in previous work, it is possible that more data would allow our approach to achieve even better performance. In the next section we will discuss the real-world value offered by our approach that is largely missing from existing approaches.

4 Actionable Insights

A primary benefit of applying PPDs to coherence-related tasks is the ability to gain deeper insights into the data. In this section, we will demonstrate the following in particular: (1) how PPDs can quickly be used to understand how the coherence of a text may be improved, (2) how the existence of multiple coherence subsections may be identified, and (3) how PPDs can allow users to locate specific types of information without reading a single word, a specific case of which is extractive summarization. For demonstrations, we will use the news article presented in Figure 4.

4.1 Improving Coherence

For a writer to improve their work, understanding the incoherence present is important. Observing the PPD sequence for the article in Figure 4 makes it easy to spot areas of potential incoherence: they occur where consecutive PPDs are significantly different (from sentences 1 to 2, 6 to 7, and 10 to 11). In this case, the writer may determine that sentence 2 is perhaps not as introductory as it should be. The predicted incoherence between sentences 10 and 11 is more interesting, and as we will see next, the writer may realize that this incoherence may be okay to retain.

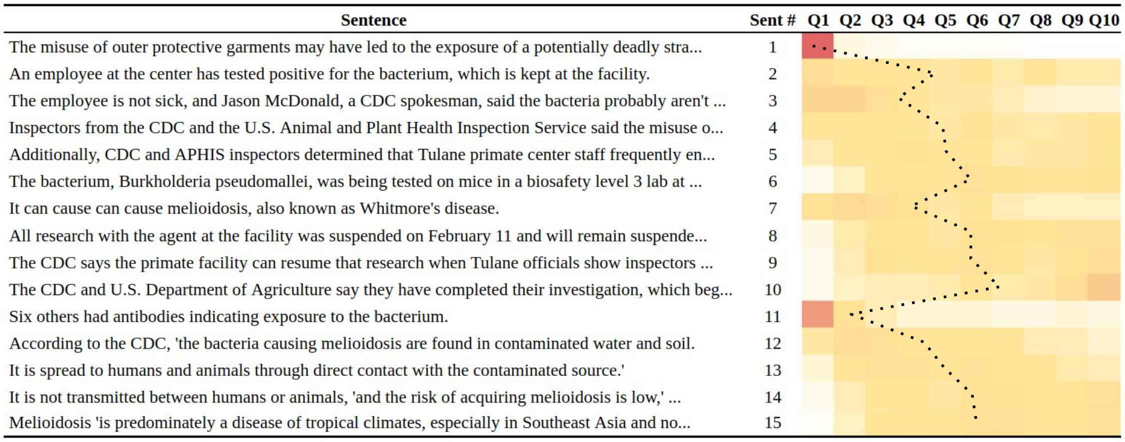
4.2 Identifying Subsections

In Figure 4, we see rough progressions of introductory-type sentences to conclusory-type sentences between sentences 1 and 10 and sentences 11 and 15. This may indicate that the article is actually composed of two coherent subsections, which means that the incoherence between sentences 10 and 11 is expected and natural. By

[3] https://www.kaggle.com/benhamner/nips-papers

	Order discrimination		Reordering	
Model	**Accident**	**Earthquake**	**Acc**	τ
Random	50	50	15.6	0
Entiry Grid	90.4	87.2	20.1	0.09
Window network	-	-	41.7	0.59
LSTM_PtrNet	93.7	99.5	50.9	0.67
RNN Decoder	-	-	48.2	0.67
Varient-LSTM+PtrNet	94.4	99.7	51.6	**0.72**
ATTOrderNet	**96.2**	**99.8**	**56.1**	**0.72**
PPDs	94.4	99.3	54.9	**0.72**

Table 2: Results on the order discrimination and sentence reordering coherence tasks. Our approach trades only a small decrease in performance for improved utility of the sentence embeddings over other approaches, achieving close to or the same as the state-of-the-art.



Sentence	Sent #
The misuse of outer protective garments may have led to the exposure of a potentially deadly stra...	1
An employee at the center has tested positive for the bacterium, which is kept at the facility.	2
The employee is not sick, and Jason McDonald, a CDC spokesman, said the bacteria probably aren't ...	3
Inspectors from the CDC and the U.S. Animal and Plant Health Inspection Service said the misuse o...	4
Additionally, CDC and APHIS inspectors determined that Tulane primate center staff frequently en...	5
The bacterium, Burkholderia pseudomallei, was being tested on mice in a biosafety level 3 lab at ...	6
It can cause can cause melioidosis, also known as Whitmore's disease.	7
All research with the agent at the facility was suspended on February 11 and will remain suspende...	8
The CDC says the primate facility can resume that research when Tulane officials show inspectors ...	9
The CDC and U.S. Department of Agriculture say they have completed their investigation, which beg...	10
Six others had antibodies indicating exposure to the bacterium.	11
According to the CDC, 'the bacteria causing melioidosis are found in contaminated water and soil.	12
It is spread to humans and animals through direct contact with the contaminated source.'	13
It is not transmitted between humans or animals, 'and the risk of acquiring melioidosis is low,' ...	14
Melioidosis 'is predominately a disease of tropical climates, especially in Southeast Asia and no...	15

Figure 4: The PPDs for a CNN article. (full text available at http://web.archive.org/web/20150801040019id_/http://www.cnn.com/2015/03/13/us/tulane-bacteria-exposure/). The dashed line shows the weighted average predicted sentence positions.

being able to understand where subsections may occur in a document, a writer can make informed decisions on where to split a long text into more coherent chunks or paragraphs. Knowing where approximate borders between ideas in a document exist may also help readers skim the document to find desired information more quickly, as further discussed in the next subsection.

4.3 Locating Information and Summarization

When reading a new article, readers well-versed in the subject of the article may want to skip high-level introductory comments and jump straight to the details. For those unfamiliar with the content or triaging many articles, this introductory information is important to determine the subject matter. Using PPDs, locating these types of information quickly should be easy for readers, even when the document has multiple potential subsections. In Figure 4, sentences 1 and 11 likely contain introductory information (since the probability of occurring in the first quantiles is highest), the most conclusory-type information is in sentence 10, and lower-level details are likely spread among the remaining sentences.

Locating sentences with the high-level details of a document is reminiscent of the task of extractive summarization, where significant research has been performed (Nenkova et al., 2011; Nenkova

Model (lead baseline source)	ROUGE-1	ROUGE-2	ROUGE-L
Lead-3 (Nallapati et al., 2017)	39.2	15.7	35.5
Lead-3 (See et al., 2017)	40.3	17.7	36.6
Lead-3 (Ours)	35.8	15.9	33.5
SummaRuNNer (Nallapati et al., 2017) ((Nallapati et al., 2017))	39.6	16.2	35.3
Pointer-generator (See et al., 2017) ((See et al., 2017))	39.5	17.3	36.4
RL (Paulus et al., 2017) ((Nallapati et al., 2017))	41.2	15.8	39.1
TextRank (Mihalcea and Tarau, 2004) (ours)	26.2	11.1	24.3
Luhn (Luhn, 1958) (ours)	26.4	11.2	24.5
SumBasic (Nenkova and Vanderwende, 2005) (ours)	27.8	10.4	26.0
LexRank (Erkan and Radev, 2004) (ours)	28.4	11.6	26.3
PPDs (ours)	**30.1**	**12.6**	**28.2**

Table 3: ROUGE scores on the CNN/DailyMail summarization task. Our PPD-based heuristic outperforms the suite of established heuristic summarizers. However, the higher performance of the deep-learning models demonstrates that training explicitly for summarization is beneficial.

and McKeown, 2012). It is thus natural to ask how well a simple PPD-based approach performs at summarization. To answer this question, the summarization algorithm we will use is: select the n sentences with the highest $PPD(s \in S)_0$ value, where S is the article being extractively summarized down to n sentences. For the article in Figure 4, sentences 1, 11, and 3 would be chosen since they have the highest first-quantile probabilities. This heuristic is conceptually similar to the Lead heuristic, where sentences that actually occur at the start of the document are chosen to be in the summary. Despite its simplicity, the Lead heuristic often achieves near state-of-the-art results (See et al., 2017).

We experiment on the non-anonymized CNN/DailyMail dataset (Hermann et al., 2015) and evaluate with full-length ROUGE-1, -2, and -L F1 scores (Lin and Hovy, 2003). For the neural position model, we choose four promising sets of hyperparameters identified during the hyperparameter search for the sentence ordering task in Section 3 and train each sentence position model on 10K of the 277K training articles (which provides our sentence position model with over 270K sentences to train on). Test results are reported for the model with the highest validation score. The final hyperparameters chosen for this sentence location model are: Q = 10, epochs = 10, layer dropouts = (0.4, 0.2), layer widths = (512, 64).

We compare our PPD-based approach to other heuristic approaches[4]. For completeness, we also include results of deep-learning based approaches and their associated Lead baselines evaluated using full-length ROUGE scores on the non-anonymized CNN/DailyMail dataset.

Table 3 contains the the comparison between our PPD-based summarizer and several established heuristic summarizers. We observe that our model has ROUGE scores superior to the other heuristic approaches by a margin of approximately 2 points for ROUGE-1 and -L and 1 point for ROUGE-2. In contrast, the deep-learning approaches trained explicitly for summarization achieve even higher scores, suggesting that there is more to a good summary than the sentences simply being introductory-like.

5 Related Work

Extensive research has been done on text coherence, motivated by downstream utility of coherence models. In addition to the applications we demonstrate in Section 4, established applications include determining the readability of a text (coherent texts are easier to read) (Barzilay and Lapata, 2008), refinement of multi-document summaries (Barzilay and Elhadad, 2002), and essay scoring (Farag et al., 2018).

Traditional methods to coherence modelling utilize established theory and handcrafted linguistic features (Grosz et al., 1995; Lapata, 2003). The Entity Grid model (Lapata and Barzilay, 2005; Barzilay and Lapata, 2008) is an influential traditional approach which works by first constructing a sentence × discourse entities (noun phrases) occurrence matrix, keeping track of the syntactic role of each entity in each sentence. Sentence tran-

[4] Implementations provided by Sumy library, available at https://pypi.python.org/pypi/sumy.

sition probabilities are then calculated using this representation and used as a feature vector as input to a SVM classifier trained to rank sentences on coherence.

Newer methods utilizing neural networks and deep learning can be grouped together by whether they indirectly or directly produce an ordering given an unordered set of sentences.

Indirect ordering. Approaches in the indirect case include Window network (Li and Hovy, 2014), Pairwise Ranking Model (Chen et al., 2016), the deep coherence model from (Cui et al., 2017), and the discriminative model from (Li and Jurafsky, 2017). These approaches are trained to take a set of sentences (anywhere from two (Chen et al., 2016) or three (Li and Hovy, 2014) to the whole text (Cui et al., 2017; Li and Jurafsky, 2017)) and predict whether the component sentences are already in a coherent order. A final ordering of sentences is constructed by maximizing coherence of sentence subsequences.

Direct ordering. Approaches in the direct case include (Gong et al., 2016; Logeswaran et al., 2018; Cui et al., 2018). These model are trained to take a set of sentences, encode them using some technique, and with a recurrent neural network decoder, output the order in which the sentences would coherently occur.

Models in these two groups all use similar high-level architectures: a recurrent or convolutional sentence encoder, an optional paragraph encoder, and then either predicting coherence from that encoding or iteratively reconstructing the ordering of the sentences. The PPD-based approaches described in Section 2 take a novel route of directly predicting location information of each sentence. Our approaches are thus similar to the direct approaches in that position information is directly obtained (here, in the PPDs), however the position information produced by our model is much more rich than simply the index of the sentence in the new ordering. With the set of indirect ordering approaches, our model approach to coherence modelling shares the property that induction of an ordering upon the sentences is only done after examining all of the sentence embeddings and explicitly arranging them in the most coherent fashion.

6 Conclusions

The ability to facilitate deeper understanding of texts is an important, but recently ignored, property for coherence modelling approaches. In an effort to improve this situation, we present a self-supervised approach to learning sentence embeddings, which we call PPDs, that rely on the connection between the meaning of a sentence and its location in a text. We implement the new sentence embedding technique with a recurrent neural network trained to map a sentence to a discrete distribution indicating where in the text the sentence is likely located. These PPDs have the useful property that a high probability in a given quantile indicates that the sentence is typical of sentences that would occur at the corresponding location in the text.

We demonstrate how these PPDs can be applied to coherence tasks with algorithms simple enough such that they can be visually performed by users while achieving near state-of-the-art, outperforming more complex and specialized systems. We also demonstrate how PPDs can be used to obtain various insights into data, including how to go about improving the writing, how to identify potential subsections, and how to locate specific types of information, such as introductory or summary information. As a proof-of-concept, we additionally show that despite PPDs not being designed for the task, they can be used to create a heuristic summarizer which outperforms comparable heuristic summarizers.

In future work, it would be valuable to evaluate our approach on texts from a wider array of domains and with different sources of incoherence. In particular, examining raw texts identified by humans as lacking coherence could be performed, to determine how well our model correlates with human judgment. Exploring how the algorithms utilizing PPDs may be refined for improved performance on the wide variety of coherence-related tasks may also prove fruitful. We are also interested in examining how PPDs may assist with other NLP tasks such as text generation or author identification.

Acknowledgments

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) through the Discovery Grants Program. NSERC invests annually over $1 billion

in people, discovery and innovation.

References

Regina Barzilay and Noemie Elhadad. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research* .

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics* 34(1):1–34.

Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. Neural sentence ordering. *arXiv preprint arXiv:1607.06952* .

François Chollet et al. 2015. Keras. https://github.com/keras-team/keras.

Baiyun Cui, Yingming Li, Ming Chen, and Zhongfei Zhang. 2018. Deep attentive sentence ordering network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 4340–4349. http://aclweb.org/anthology/D18-1465.

Baiyun Cui, Yingming Li, Yaqing Zhang, and Zhongfei Zhang. 2017. Text coherence analysis based on deep neural network. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, pages 2027–2030.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research* 22:457–479.

Youmna Farag, Helen Yannakoudakis, and Ted Briscoe. 2018. Neural automated essay scoring and coherence modeling for adversarially crafted input. *arXiv preprint arXiv:1804.06898* .

Jingjing Gong, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. End-to-end neural sentence ordering using pointer network. *arXiv preprint arXiv:1611.04953* .

Barbara J Grosz, Scott Weinstein, and Aravind K Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational linguistics* 21(2):203–225.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. pages 1693–1701.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* .

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Mirella Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, pages 545–552.

Mirella Lapata and Regina Barzilay. 2005. Automatic evaluation of text coherence: Models and representations. In *IJCAI*. volume 5, pages 1085–1090.

Jiwei Li and Eduard Hovy. 2014. A model of coherence based on distributed sentence representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. pages 2039–2048.

Jiwei Li and Dan Jurafsky. 2017. Neural net models of open-domain discourse coherence. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. pages 198–209.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram cooccurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, pages 71–78.

Lajanugen Logeswaran, Honglak Lee, and Dragomir Radev. 2018. Sentence ordering and coherence modeling using recurrent neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of research and development* 2(2):159–165.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*. pages 3075–3081.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining text data*, Springer, pages 43–76.

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

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

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304* .

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

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. volume 1, pages 1073–1083.

Simone Teufel et al. 1999. *Argumentative zoning: Information extraction from scientific text*. Ph.D. thesis, Citeseer.

Chapter 4

Paper 2

A version of this paper, titled “Catching Attention with Automatic Pull Quote Selection”, was submitted to and published by the International Conference on Computational Linguistics (COLING) 2020. It was accepted for oral presentation and publication in the conference proceedings [8].

This paper presents a new task called “pull quote selection”, which can help understand situational interestingness (i.e. things that attract the attention of readers in general). By being able to automatically select good pull quotes to place inside an article, it stands to reason that the situational interestingness of the article can be improved. Human evaluation supports the effectiveness of our selection models, with interestingness being comparable to pull quotes occurring in published news articles.

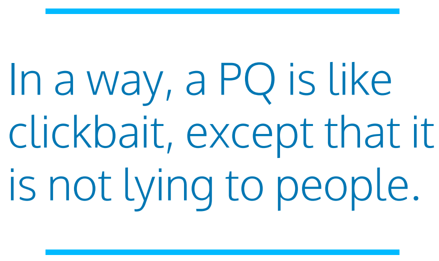
Catching Attention with Automatic Pull Quote Selection

Abstract

To advance understanding on how to engage readers, we advocate the novel task of automatic pull quote selection. Pull quotes are a component of articles specifically designed to catch the attention of readers with spans of text selected from the article and given more salient presentation. This task differs from related tasks such as summarization and clickbait identification by several aspects. We establish a spectrum of baseline approaches to the task, ranging from handcrafted features to a neural mixture-of-experts to cross-task models. By examining the contributions of individual features and embedding dimensions from these models, we uncover unexpected properties of pull quotes to help answer the important question of what engages readers. Human evaluation also supports the uniqueness of this task and the suitability of our selection models. The benefits of exploring this problem further are clear: pull quotes increase enjoyment and readability, shape reader perceptions, and facilitate learning. Code to reproduce this work is available at `https://github.com/tannerbohn/AutomaticPullQuoteSelection`.

1 Introduction

Discovering what keeps readers engaged is an important problem. We thus propose the novel task of automatic pull quote (PQ) selection accompanied with a new dataset and insightful analysis of several motivated baselines. PQs are graphical elements of articles with thought provoking spans of text pulled from an article by a writer or copy editor and presented on the page in a more salient manner (French, 2018), such as in Figure 1.



In a way, a PQ is like clickbait, except that it is not lying to people.

Figure 1: A pull quote from this paper chosen with the help of our best performing model (see Section 5.3).

PQs serve many purposes. They provide temptation (with unusual or intriguing phrases, they make strong entrypoints for a browsing reader), emphasis (by reinforcing particular aspects of the article), and improve overall visual balance and excitement (Stovall, 1997; Holmes, 2015). PQ frequency in reading material is also significantly related to information recall and student ratings of enjoyment, readability, and attractiveness (Wanta and Gao, 1994; Wanta and Remy, 1994).

The problem of automatically selecting PQs is related to the previously studied tasks of headline success prediction (Piotrkowicz et al., 2017; Lamprinidis et al., 2018), clickbait identification (Potthast et al., 2016; Chakraborty et al., 2016; Venneti and Alam, 2018), as well as key phrase extraction (Hasan and Ng, 2014) and document summarization (Nenkova and McKeown, 2012). However, in Sections 5.4 and 5.5 we provide experimental evidence that performing well on these previous tasks does not translate to performing well at PQ selection. Each of these types of text has a different function in the context of engaging a reader. The title tells the reader what the article is about and sets the tone. Clickbait makes unwarranted enticing promises of what the article is about. Key phrases and summaries help the reader decide whether the topic is of interest. And PQs provide *specific* intriguing entrypoints for the reader or can *maintain* interest once reading has begun by providing glimpses of interesting things to come. With their unique qualities, we believe PQs satisfy important roles missed by these popular existing tasks.

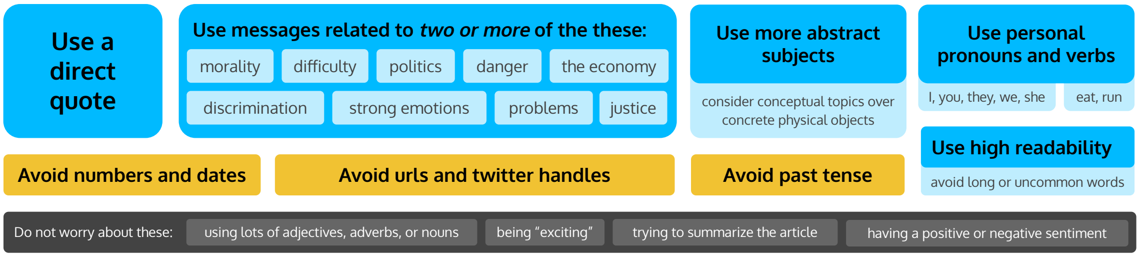

Use a direct quote
Use messages related to *two or more* of the these:
morality
difficulty
politics
danger
the economy
discrimination
strong emotions
problems
justice
Use more abstract subjects
consider conceptual topics over concrete physical objects
Use personal pronouns and verbs
I, you, they, we, she
eat, run
Use high readability
avoid long or uncommon words
Avoid numbers and dates
Avoid urls and twitter handles
Avoid past tense
Do not worry about these:
using lots of adjectives, adverbs, or nouns
being "exciting"
trying to summarize the article
having a positive or negative sentiment

Figure 2: Factors suggested by our results to be important (and unimportant) in creating pull quotes.

In this work we define PQ selection as a sentence classification task and create a dataset of articles and their expert-selected PQs from a variety of news sources. We establish a number of approaches with which to solve and gain insight into this task: (1) handcrafted features, (2) n-gram encodings, (3) Sentence-BERT (SBERT) (Reimers and Gurevych, 2019) embeddings combined with a progression of neural architectures, and (4) cross-task models. Via each of these model groups, we uncover interesting patterns (summarized in Figure 2). For example, among handcrafted features, sentiment and arousal are surprisingly uninformative features, overshadowed by presence of quotation marks and reading difficulty. Analysing individual SBERT embedding dimensions also helps understand the particular themes that make for a good PQ. We also find that combining SBERT sentence and document embeddings in a mixture-of-experts manner provide the best performance at PQ selection. The suitability of our models at PQ selection is also supported via human evaluation.

The main contributions are:

1. We describe several motivated approaches for the new task of PQ selection, including a mixture-of-experts approach to combine sentence and document embeddings (Section 3).

2. We construct a dataset for training and evaluation of automatic PQ selection (Section 4).

3. We inspect the performance of our approaches to gain a deeper understanding of PQs, their relation to other tasks, and what engages readers (Section 5). Figure. 2 summarizes these findings.

2 Related Work

In this section, we look at three natural language processing tasks related to PQ selection: (1) headline quality prediction, (2) clickbait identification, and (3) summarization and keyphrase extraction. These topics motivate the cross-task models whose performance on PQ selection is reported in Section 5.4.

2.1 Headline Quality Prediction

When a reader comes across a news article, the headline is often the first thing given a chance to catch their attention, thus predicting their success is a strongly motivated task. Once a reader decides to check out the article, it is up to the content (including PQs) to maintain their engagement.

In (Piotrkowicz et al., 2017), the authors experimented with two sets of features: journalism-inspired (which aim to measure how news-worthy the topic itself is), and linguistic style features (reflecting properties such as length, readability, and parts-of-speech – we consider such features here). They found that overall the simpler style features work better than the more complex journalism-inspired features at predicting social media popularity of news articles. The success of simple features is also reflected in (Lamprinidis et al., 2018), which proposed multi-task training of a recurrent neural network to not only predict headline popularity given pre-trained word embeddings, but also predict its topic and parts-of-speech tags. They found that while the multi-task learning helped, it performed only as well as a logistic regression model using character n-grams. Similar to these previous works, we also evaluate several expert-knowledge based features and n-grams, however, we expand upon this to include a larger variety of models and provide a more thorough inspection of performance to understand what engages readers.

2.2 Clickbait Identification

The detection of a certain type of headline – clickbait – is a recently popular task of study. Clickbait is a particularly catchy headline and form of false advertising used by news outlets which lure potential readers but often fail to meet expectations, leaving readers disappointed (Potthast et al., 2016). Clickbait examples include "You Won't Believe..." or "X Things You Should...". We suspect that the task of distinguishing between clickbait and non-clickbait headlines is related to PQ selection because both tasks may rely on identifying the catchiness of a span of text. However, PQs attract your attention with content truly in the article. In a way, a PQ is like clickbait, except that it is not lying to people.

In (Venneti and Alam, 2018), the authors found that measures of topic novelty (estimated using LDA) and surprise (based on word bi-gram frequency) were strong features for detecting clickbait. In our work however, we investigate the interesting topics themselves (Section 5.3). A set of 215 handcrafted features were considered in (Potthast et al., 2016) including sentiment, length statistics, specific word occurrences, but the authors found that the most successful features were character and word n-grams. The strength of n-gram features at this task is also supported by (Chakraborty et al., 2016). While we also demonstrate the surprising effectiveness of n-grams and consider a variety of handcrafted features for our particular task, we examine more advanced approaches that exhibit superior performance.

2.3 Summarization and Keyphrase Extraction

Document summarization and keyphrase extraction are two well-studied NLP tasks with the goals of capturing and conveying the main topics and key information discussed in a body of text (Turney, 1999; Nenkova and McKeown, 2012). Keyphrase extraction is concerned with doing this at the level of individual phrases, while extractive document summarization (which is just one type of summarization (Nenkova et al., 2011)) aims to do this at the sentence level. Approaches to summarization have roughly evolved from unsupervised extractive heuristic-based methods (Luhn, 1958; Mihalcea and Tarau, 2004; Erkan and Radev, 2004; Nenkova and Vanderwende, 2005; Haghighi and Vanderwende, 2009), to supervised and often abstractive deep-learning approaches (Nallapati et al., 2016b; Nallapati et al., 2016a; Nallapati et al., 2017; Zhang et al., 2019). Approaches to keyphrase extraction fall into similar groups, with unsupervised approaches including (Tomokiyo and Hurst, 2003; Mihalcea and Tarau, 2004; Liu et al., 2009), and supervised approaches including (Turney, 1999; Medelyan et al., 2009; Romary, 2010).

While summarization and keyphrase extraction are concerned with what is *important* or representative in a document, we instead are interested in understanding what is *engaging*. While these two concepts may seem very similar, in Sections 5.4 and 5.4 we provide evidence of their difference by demonstrating that what makes for a good summary does not make for a good PQ.

3 Models

We consider four groups of approaches for the PQ selection task: (1) handcrafted features (Section 3.1), (2) n-gram features (Section 3.2), (3) SBERT embeddings combined with a progression of neural architectures (Section 3.3), and (4) cross-task models (Section 3.4). As discussed further in Section 4, these approaches aim to determine the probability that a given article sentence will be used for a PQ.

3.1 Handcrafted Features

Our handcrafted features can be loosely grouped into three categories: surface, parts-of-speech, and affect, each of which we will provide justification for. For the classifier we will use AdaBoost (Hastie et al., 2009) with a decision tree base estimator, as this was found to outperform simpler classifiers without requiring much hyperparameter tuning.

3.1.1 Surface Features

- **Length**: We expect that writers have a preference to choose PQs which are concise. To measure length, we will use the total character length, as this more accurately reflects the space used by the text than the number of words.

- **Sentence position**: We consider the location of the sentence in the document (from 0 to 1). This is motivated by the finding in summarization that summary-suitable sentences tend to occur near the beginning (Braddock, 1974) – perhaps a similar trend exists for PQs.

- **Quotation marks**: We observe that PQs often contain content from direct quotations. As a feature, we thus include the count of opening and closing double quotation marks.

- **Readability**: Motivated by the assumption that writers will not purposefully choose difficult-to-read PQs, we consider two readability metric features: (1) **Flesch Reading Ease**: This measure (R_{Flesch}) defines reading ease in terms of the number of words per sentence and the number of syllables per word (Flesch, 1979). (2) **Difficult words**: This measure ($R_{difficult}$) is the percentage of unique words which are considered "difficult" (at least six characters long and not in a list of $\sim$3000 easy-to-understand words). See Appendix A for details.

3.1.2 Part-of-Speech Features

We include the word density of part-of-speech (POS) tags in a sentence as a feature. As suggested by (Piotrkowicz et al., 2017) with respect to writing good headlines, we suspect that verb (VB) and adverb (RB) density will be informative. We also report results on the following: cardinal digit (CD), adjective (JJ), modal verb (MD), singular noun (NN), proper noun (NNP), personal pronoun (PRP).

3.1.3 Affect Features

Events or images that are shocking, filled with emotion, or otherwise exciting will attract attention (Schupp et al., 2007). However, this does not necessarily mean that text describing these things will catch reader interest as reliably (Aquino and Arnell, 2007). To determine how predictive sentence affect properties are of PQ suitability, we include the following features:

Positive sentiment (A_{pos}) and **negative sentiment**(A_{neg}).

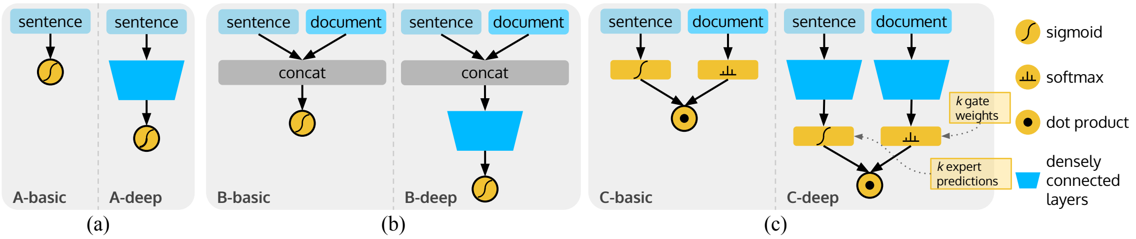
Compound sentiment ($A_{compound}$). This combines the positive and negative sentiments to represent overall sentiment between -1 and 1.

Valence ($A_{valence}$) and **arousal** ($A_{arousal}$): Valence refers to the pleasantness of a stimulus and arousal refers to the intensity of emotion provoked by a stimulus (Warriner et al., 2013). In (Aquino and Arnell, 2007), the authors specifically note that it is the arousal level of words, and not valence which is predictive of their effect on attention (measured via reaction time). Measuring early cortical responses and recall, (Kissler et al., 2007) observed that words of greater valence were both more salient and memorable. To measure valence and arousal of a sentence, we use the averaged word rating, utilizing word ratings from the database introduced by (Warriner et al., 2013).

Concreteness ($A_{concreteness}$): This is "the degree to which the concept denoted by a word refers to a perceptible entity" (Brysbaert et al., 2014). As demonstrated by (Sadoski et al., 2000), concrete texts are better recalled than abstract ones and concreteness is a strong predictor of text comprehensibility, interest, and recall. To measure concreteness of a sentence, we use the averaged word rating, utilizing word ratings in the database introduced by (Brysbaert et al., 2014).

3.2 N-Gram Features

We consider character-level and word-level n-gram text representations, shown to perform well in related tasks (Potthast et al., 2016; Chakraborty et al., 2016; Lamprinidis et al., 2018). A passage of text is then represented by a vector of the counts of the individual n-grams it contains. We use a logistic regression classifier with these representations with $L2$ regularization and an inverse-regularization strength of 1.



(a) (b) (c)

Figure 3: The progression of neural network architectures combined with SBERT sentence and document embeddings. Group A only uses sentence embeddings, while groups B and C also use document embeddings. In group C, they are combined in a mixture-of-experts fashion (the width of the sigmoid and softmax layers is equal to the # experts). For each group, there is a basic version and deep version.

3.3 SBERT Embeddings with a Progression of Neural Architectures

All other models described in this work use only the single sentence to predict PQ probability. To understand the importance of considering the entire article when choosing PQs, we consider three groups of neural architectures, as shown in Figure 3.

Group A. These neural networks only take the sentence embedding as input. In the **A-basic** model, there are no hidden layers. In **A-deep**, the embedding passes through a set of densely connected layers.

Group B. These models receive the sentence embedding and a whole-document embedding as input. This allows the models to account for document-dependent patterns. These embeddings are concatenated and connected to the output node (**B-basic**), or first pass through densely connected layers (**B-deep**).

Group C. These networks also receive sentence and document embeddings, but they are combined in a mixture-of-experts manner (Jacobs et al., 1991). That is, multiple predictions are produced by a set of "experts" and a gating mechanism determines the weighting of these predictions for a given input. The motivation is that there may be many "types" of articles, each requiring paying attention to different properties when choosing a PQ. If each of k experts generates a prediction, we can use the document embedding to determine the weighting over the predictions. In Figure 3c, k corresponds to the width of the sigmoid and softmax layers, which are then combined with a dot product to produce the final prediction. In **C-deep**, the embeddings first pass through a set of densely connected layers (non-shared weights) as shown in the right of Figure 3c, while in **C-basic**, they do not.

To embed sentences and documents, we make use of a pre-trained Sentence-BERT (SBERT) model (Reimers and Gurevych, 2019). SBERT is a modification of BERT (Bidirectional Encoder Representations from Transformers) – a language representation model which performs well on a wide variety of tasks (Devlin et al., 2018). SBERT is designed to more efficiently produce semantically meaningful embeddings (Reimers and Gurevych, 2019). We computed document embeddings by averaging SBERT sentence embeddings.

3.4 Cross-Task Models

To test the similarity of PQ selection with related tasks , we use the following models: **Headline popularity**: We train a model to predict the popularity of a headline (using SBERT embeddings and linear regression) with the dataset introduced by (Moniz and Torgo, 2018). This dataset includes feedback metrics for about 100K news articles from various social media platforms. We apply this model to PQ selection by predicting the popularity of each sentence, scaling the predictions for each article to lie in $[0, 1]$ and interpreting these values as PQ probability. **Clickbait identification**: We train a model to discriminate between clickbait and non-clickbait headlines (using SBERT embeddings and logistic regression) with the dataset introduced by (Chakraborty et al., 2016). Clickbait probability is used as a proxy for PQ probability. **Summarization**: Using a variety of extractive summarizers, we score each sentence in an article, scale the values to lie in $[0, 1]$, and interpret these values as PQ probability. No training is required for this model. Appendix. A contain implementation details of these models

4 Experimental Setup

To support the new task of automatic PQ selection, we both construct a new dataset and describe a suitable evaluation metric.

4.1 Datatset Construction

To conduct our experiments, we create a dataset using articles from several online news outlets: National Post, The Intercept, Ottawa Citizen, and Cosmopolitan. For each outlet, we identify those articles containing at least one pull quote. From these articles, we extract the *body*, *edited PQs*, and *PQ source sentences*. The body contains the full list of sentences composing the body of the article. The edited PQs are the pulled texts as they appear after being augmented by the editor to appear as pull quotes[1]. The PQ source sentences are the article sentences from which the edited PQs came. In this work, we aim to determine whether a given article sentence is a source sentence or not[2].

Dataset statistics are reoprted in Table 1. It contains ∼27K positive samples (PQ source sentences—which we simply call PQ sentences) and ∼680K negative samples (non-PQ sentences). The positive to negative ratio is 1:26 (taken into consideration when training our classifiers with balanced class weights). For all experiments, we use the same training/validation/test split of the articles (70/10/20).

	nationalpost	**theintercept**	**ottawacitizen**	**cosmopolitan**	**train**	**val**	**test**	**all**
# articles	11112	1183	1066	1267	10239	1462	2927	14628
# PQ	16307	2671	1087	2360	15709	2235	4481	22425
# PQ/article	1.47	2.26	1.02	1.86	1.53	1.53	1.53	1.53
# sentences/PQ	1.16	1.23	1.32	1.24	1.19	1.18	1.19	1.19
# sentences/article	40.49	97.94	38.35	79.03	48.47	47.8	48.06	48.32
# pos samples	18975	3274	1436	2906	18640	2625	5326	26591
# neg samples	430959	112588	39443	97230	477609	67258	135353	680220

Table 1: Statistics of our PQ dataset, composed of articles from four different news outlets. Only articles with at least one PQ are included in the dataset.

4.2 Evaluation

What do we want to measure? We want to evaluate a PQ selection model on its ability to determine which sentences are more likely to be chosen by an expert as PQ source sentences.

Metric. We will use the probability that a random PQ source sentence is scored by the model above a random non-source sentence from the same article (i.e. AUC). Let $a_{inclusions}$ be the binary vector indicating whether each sentence of article a is truly a PQ source sentence, and let $\hat{a}_{inclusions}$ be the corresponding predicted probabilities. Our metric can then be computed with Equation 1, which computes the AUC averaged across articles.

$$AUC_{avg} = \frac{1}{\#articles} \sum_{a \in articles} AUC(a_{inclusions}, \hat{a}_{inclusions}) \tag{1}$$

Why average across articles? By averaging scores for each article instead of for all sentences at the same time, the evaluation method accounts for the observation that some articles may be more "pull-quotable" than others. If articles are instead combined when computing AUC, an average sentence from an interesting article can be ranked higher than the best sentence from a less interesting article.

5 Experimental Results

We present our experimental results and analysis for the four groups of approaches: handcrafted features (Section 5.1), n-gram features (Section 5.2), SBERT embeddings combined with a progression of

[1] This can include replacing pronouns such as "she", "they", "it", with the more precise nouns or proper nouns, or shortening sentences by removing individual words or clauses, or even replacing words with ones of a similar meaning but different length in order to achieve a clean text rag.

[2] A PQ source sentence could be only part of a multi-sentence PQ or contain the PQ inside it.

neural architectures (Section 5.3), and cross-task models (Section 5.4). We also perform human evaluation of several models (Section 5.5). Appendix A contains implementation details of our models, and Appendix C includes examples of PQ sentences selected by several models on various articles.

5.1 Handcrafted Features

The performance of each of our handcrafted features is provided in Figure 4a. There are several interesting observations, including some that support and contradict hypotheses made in Section 3.1:

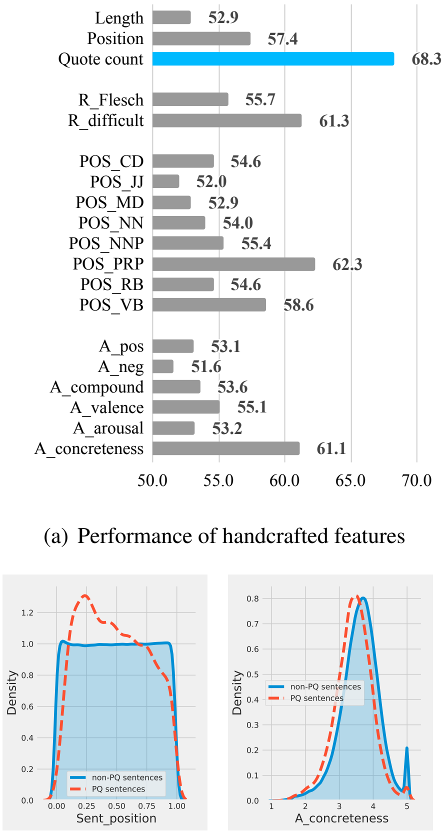
Sentence position. Simply using the sentence position works better than random guessing. When we inspect the distribution of this feature value for PQ and non-PQ sentences in Figure 4b, we see that PQ sentences are not uniformly distributed throughout articles, but rather tend to occur slightly more often around a quarter of the way through the article.

Quotation mark count.. The number of quotation marks is by far the best feature in this group, confirming that direct quotations make for good PQs. We find that a given non-PQ sentence is ∼3 times more likely not to contain quotation marks than a PQ sentence.

Reading difficulty. The fraction of difficult words is the third-best handcrafted feature, outperforming the Flesch metric. As suggested in Section 3.1.1 we find that PQ sentences are indeed easier to read than non-PQ sentences.

POS tags. Of the POS tag densities, personal pronoun (PRP) and verb (VB) density are the most informative. Inspecting the feature distributions, we see that PQs tend to have slightly higher PRP density as well as VB density – suggesting that sentences about people doing things are good candidates for PQs.

Affect features. Affect features tended to perform poorly, contradicting our intuition that more exciting or emotional sentences would be chosen for PQs. However, concreteness is indeed an informative feature, with *decreased* concreteness unexpectedly being better (see Figure 4c). Given the memorability that comes with more concrete texts (Sadoski et al., 2000), this suggests that something else may be at work in order to explain the beneficial effects of PQs on learning outcomes (Wanta and Gao, 1994; Wanta and Remy, 1994).



(a) Performance of handcrafted features

(b) Sentence position

(c) Concreteness

Figure 4: The value distributions for two interesting handcrafted features for both non-PQ sentences (solid blue region) and PQ sentences (dashed orange lines).

5.2 N-Gram Features

The results for our n-gram models are provided in Table 2. Impressively, almost all n-gram models performed better than any individual handcrafted feature, with the best model, character bi-grams, demonstrating an AUC_{avg} of 75.4. When we inspect the learned logistic regression weights for the best variant of each model type (summarized in Figure 5), we find a few interesting observations:

Top character bi-grams. The highest weighted character bi-grams exclusively aim to identify the beginnings of quotations, agreeing with the success of the quote count feature that the presence of a quote is highly informative. Curiously, the presence of a quotation being present but not starting the sentence is a strong negative indicator (i.e. “ “”).

Bottom character bi-grams. Among the lowest weighted character bi-grams are also indicators of numbers, URLs, and possibly twitter handles (i.e. “@”).

Token	n = 1	n = 2	n = 3
char	70.7	75.4	74.2
word	73.9	72.3	65.6

Table 2: AUC_{avg} scores of the n-gram models.

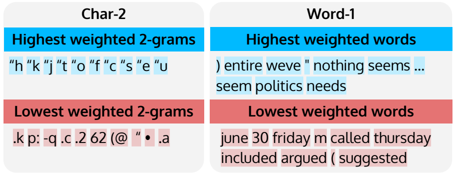

Char-2
Highest weighted 2-grams
“h “k “j “t “o “f “c “s “e “u
Lowest weighted 2-grams
.k p: -q .c .2 62 (@ “ • .a
Word-1
Highest weighted words
) entire weve " nothing seems ... seem politics needs
Lowest weighted words
june 30 friday m called thursday included argued (suggested

Figure 5: The ten highest and lowest weighted n-grams for the best character and word models.

Words. Although the highest weighted words are difficult to interpret together, among the lowest weighted words are those indicating past tense: “called”, “included”, “argued”, “suggested”. This suggests a promising approach for PQ selection includes identification of the tense of each sentence.

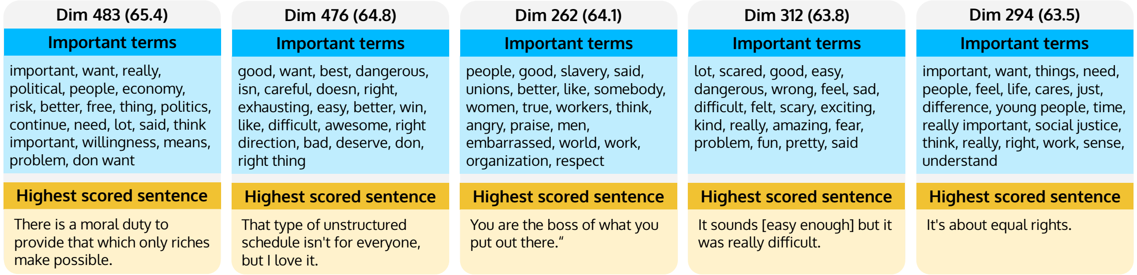
5.3 SBERT Embeddings with a Progression of Neural Architectures

The results of the neural architectures using SBERT embeddings is included in Table 3. Overall, these results suggest that using document embeddings helps performance, especially with a mixture-of-experts architecture. This is seen by the general trend of improved performance from group A to B to C. Within each group, adding the fully connected layers (the “deep” models) helps.

Model	AUC_{avg}	**Width**	**# Params**
A-basic	76.7±0.15	-	7.7E+02
A-deep	77.7±0.16	128, 64	1.1E+05
B-basic	77.1±0.24	-	1.5E+03
B-deep	78.3±0.29	128, 64	2.1E+05
C-basic ($k = 16$)	77.7±0.51	-	2.5E+04
C-deep ($k = 4$)	78.7±0.07	32, 16	5.0E+04

Table 3: Results on the neural architectures. Performance mean and std. dev. is calculated with five trials. k refers to the # experts, only applicable to C group models. Width values correspond to the width of the two additional fully connected layers (only applicable to the deep models).

Inspecting individual SBERT dimensions. Given the performance of these embeddings, we are eager to understand what aspects of the text it picks up on. To do this, we first identify the most informative of the 768 dimensions for PQ selection by training a logistic regression model for each one. For each single-feature model, we group sentences in the test set by PQ probability (high, medium, and low) and perform a TF-IDF analysis to identify key terms associated with *increasing* PQ probability[3]. See Appendix B for more details. Results for the top five best performing dimensions are shown in Figure 6. We find that each of these dimension is sensitive to the presence of a theme (or combination of themes) generally interesting and important to society. Our interpretations of them are: (a) politics and doing the right thing, (b) working hard on difficult/dangerous things, (c) discrimination, (d) strong emotions – both positive and negative, and (e) social justice.


Dim 483 (65.4)
Important terms
important, want, really, political, people, economy, risk, better, free, thing, politics, continue, need, lot, said, think important, willingness, means, problem, don want
Highest scored sentence
There is a moral duty to provide that which only riches make possible.

(a)

Dim 476 (64.8)
Important terms
good, want, best, dangerous, isn, careful, doesn, right, exhausting, easy, better, win, like, difficult, awesome, right direction, bad, deserve, don, right thing
Highest scored sentence
That type of unstructured schedule isn't for everyone, but I love it.

(b)

Dim 262 (64.1)
Important terms
people, good, slavery, said, unions, better, like, somebody, women, true, workers, think, angry, praise, men, embarrassed, world, work, organization, respect
Highest scored sentence
You are the boss of what you put out there.”

(c)

Dim 312 (63.8)
Important terms
lot, scared, good, easy, dangerous, wrong, feel, sad, difficult, felt, scary, exciting, kind, really, amazing, fear, problem, fun, pretty, said
Highest scored sentence
It sounds [easy enough] but it was really difficult.

(d)

Dim 294 (63.5)
Important terms
important, want, things, need, people, feel, life, cares, just, difference, young people, time, really important, social justice, think, really, right, work, sense, understand
Highest scored sentence
It's about equal rights.

(e)

Figure 6: The top five best performing SBERT embedding dimensions, along with the terms associated with increasing PQ probability with respect to that dimension. For each dimension, we also include the sentence from the test articles which that dimension most strongly scores as being a PQ sentence. At the top of each box is the dimension index and the test AUC_{avg}.

[3] Likewise, we could study terms associated with *decreasing* PQ probability – to deeper understand what *bores* people.

5.4 Cross-Task Models

The results for the cross-task models of headline popularity prediction, clickbait identification, and summarization are shown in Table 4. Considered holistically, the results suggest that PQs are not designed to inform the reader about what they are reading (the shared purpose of headlines and summaries), so much as they are designed to motivate further engagement (the sole purpose of clickbait). However, the considerable performance gap between the clickbait model and PQ-specific models (such as character bi-grams and SBERT embeddings) suggest that this is only one aspect of choosing good pull quotes.

Another interesting observation is the variability in performance of summarizers at PQ selection. If we consider the summarization performance of these models as reported together in (Chen et al., 2016), we find that PQ selection performance is not strongly correlated with their summarization performance.

Model	AUC_{avg}
headline popularity	56.9
clickbait	63.8
LexRank	51.9
SumBasic	44.9
KLSum	55.1
TextRank	55.9

Table 4: Performance of the cross-task models.

5.5 Human Evaluation

As a final experiment, we conduct a qualitative evaluation to find out how well the PQs selected by various models (including the true PQ sources) compare. The results are summarized in Table 5. We randomly select 50 articles from the test set and ask nine volunteers to evaluate the candidate PQs extracted by six different models. They are asked to rate each of the 300 candidate PQs based on how interested it makes them in reading more of the article on a scale of 1 (not at all interested) to 5 (very interested). For each model we report the following metrics: (1) the **rating** averaged across all responses (with 5 being the best), (2) the average **rank** within an article (with 1 being the best), and (3) **1st Place Pct.** – how often the model produces the best PQ for an article (with 100% being the best).

Model	**Rating ↑**	**Rank ↓**	**1st Place Pct. ↑**
True PQ Source	2.75	3.04	**28%**
Char-2	**2.86**	**2.74**	**28%**
C-deep	2.75	3.08	18%
Headline pop.	2.57	3.66	8%
Clickbait	2.70	3.26	18%
TextRank	2.69	3.32	14%

Table 5: The results of human evaluation comparing models in terms of how interested the reader is in reading more of the article. The ↑ and ↓ indicate whether better values for a metric are respectively higher or lower.

The results in Table 5 show that the two PQ-specific approaches (Char-2 and C-deep using the best hyperparameters from Section 5.3) perform on par or slightly better than the true PQ sources. By generally out-performing the transfer models, this further supports our claim that the PQ selection task serves a unique purpose. When looking at how often each model scores 1^{st} place, which accentuates their performance differences, we can see that the headline and summarization models in particular perform poorly. Mirroring the results from Section 5.4, among the cross-task models, the clickbait model seems to perform best.

6 Conclusion

In this work we proposed the novel task of automatic pull quote selection as a means to better understand how to engage readers. To lay foundation for the task, we created a PQ dataset and described and benchmarked four groups of approaches: handcrafted features, n-grams, SBERT-based embeddings combined with a progression of neural architectures, and cross-task models. By inspecting results, we encountered multiple curious findings to inspire further research on PQ selection and understanding reader engagement.

There are many interesting avenues for future research with regard to pull quotes. In this work we assume that all true PQs in our dataset are of equal quality, however, it would be valuable to know the quality of individual PQs. It would also be interesting to study how to make a given phrase more PQ-worthy while maintaining the original meaning. When determining the similarity of PQ selection to related tasks, it would also be worth considering alternative methods, such as applying our PQ-specific models to the related tasks instead. Additionally, to get a better understanding of the performance gap we should expect between PQ-selection and other tasks, we should first consider how well PQ models trained on one news source generalize to other news sources.

Acknowledgements

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) through the Discovery Grants Program. NSERC invests annually over $1 billion in people, discovery and innovation.

References

Jennifer M Aquino and Karen M Arnell. 2007. Attention and the processing of emotional words: Dissociating effects of arousal. *Psychonomic Bulletin & Review*, 14(3):430–435. 4

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. " O'Reilly Media, Inc.". 12

Richard Braddock. 1974. The frequency and placement of topic sentences in expository prose. *Research in the Teaching of English*, 8(3):287–302. 4

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior Research Methods*, 46(3):904–911. 4, 12

Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. 2016. Stop clickbait: Detecting and preventing clickbaits in online news media. In *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*, pages 9–16. IEEE. 1, 3, 4, 5, 13

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. 2016. Distraction-based neural networks for document summarization. *arXiv preprint arXiv:1610.08462*. 9

François Chollet et al. 2015. Keras. `https://keras.io`. 13

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. 5

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479. 3, 13

Rudolf Flesch. 1979. *How to Write Plain English: A Book for Lawyers and Consumers*. Harper & Row New York, NY. 4

Nigel French. 2018. *InDesign Type: Professional Typography with Adobe InDesign*. Adobe Press. 1

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370. 3, 13

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1262–1273, Baltimore, Maryland, June. Association for Computational Linguistics. 1

Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. 2009. Multi-class adaboost. *Statistics and its Interface*, 2(3):349–360. 3

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*. 13

Tim Holmes. 2015. *Subediting and Production for Journalists: Print, Digital & Social*. Routledge. 1

Clayton J Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International AAAI Conference on Weblogs and Social Media*. 12

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87. 5

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. 13

Johanna Kissler, Cornelia Herbert, Peter Peyk, and Markus Junghofer. 2007. Buzzwords: Early cortical responses to emotional eords during reading. *Psychological Science*, 18(6):475–480. 4

Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. 2017. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, pages 971–980. 13

Sotiris Lamprinidis, Daniel Hardt, and Dirk Hovy. 2018. Predicting news headline popularity with syntactic and semantic knowledge using multi-task learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 659–664. 1, 2, 4

Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 257–266. Association for Computational Linguistics. 3

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165. 3

Olena Medelyan, Eibe Frank, and Ian H Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1318–1327. Association for Computational Linguistics. 3

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411. 3, 13

Nuno Moniz and Luís Torgo. 2018. Multi-source social feedback of online news feeds. *arXiv preprint arXiv:1801.07055*. 5, 13

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016a. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*. 3

Ramesh Nallapati, Bowen Zhou, and Mingbo Ma. 2016b. Classify or select: Neural architectures for extractive document summarization. *arXiv preprint arXiv:1611.04244*. 3

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*. 3

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76. Springer. 1, 3

Ani Nenkova and Lucy Vanderwende. 2005. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005*, 101. 3, 13

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

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830. 13, 14

Alicja Piotrkowicz, Vania Dimitrova, Jahna Otterbacher, and Katja Markert. 2017. Headlines matter: Using headlines to predict the popularity of news articles on twitter and facebook. In *Eleventh International AAAI Conference on Web and Social Media*. 1, 2, 4

Martin Potthast, Sebastian Köpsel, Benno Stein, and Matthias Hagen. 2016. Clickbait detection. In *European Conference on Information Retrieval*, pages 810–817. Springer. 1, 3, 4

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*. 2, 5, 12

Patrice Lopez Laurent Romary. 2010. Automatic key term extraction from scientific articles in grobid. In *SemEval 2010 Workshop*, page 4. 3

Mark Sadoski, Ernest T Goetz, and Maximo Rodriguez. 2000. Engaging texts: Effects of concreteness on comprehensibility, interest, and recall in four text types. *Journal of Educational Psychology*, 92(1):85. 4, 7

Harald T Schupp, Jessica Stockburger, Maurizio Codispoti, Markus Junghöfer, Almut I Weike, and Alfons O Hamm. 2007. Selective visual attention to emotion. *Journal of Neuroscience*, 27(5):1082–1089. 4

James Glen Stovall. 1997. *Infographics: A Journalist's Guide*. Allyn & Bacon. 1

Takashi Tomokiyo and Matthew Hurst. 2003. A language model approach to keyphrase extraction. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 33–40. 3

Peter Turney. 1999. Learning to extract key phrases from text, nrc technical report erb· 1057. Technical report, Canada: National Research Council. 3

Lasya Venneti and Aniket Alam. 2018. How curiosity can be modeled for a clickbait detector. *arXiv preprint arXiv:1806.04212*. 1, 3

Wayne Wanta and Dandan Gao. 1994. Young readers and the newspaper: Information recall and perceived enjoyment, readability, and attractiveness. *Journalism Quarterly*, 71(4):926–936. 1, 7

Wayne Wanta and Jay Remy. 1994. Information recall of four newspaper elements among young readers. 1, 7

Amy Beth Warriner, Victor Kuperman, and Marc Brysbaert. 2013. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior Research Methods*, 45(4):1191–1207. 4, 12

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. 2019. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*. 3

Appendix A Implementation Details

Here we outline the various tools, datasets, and other implementation details related to our experiments:

- To perform part-of-speech tagging for feature extraction, we use the NLTK 3.4.5 perceptron tagger (Bird et al., 2009).
- To compute sentiment, the VADER Sentiment Analysis tool is used (Hutto and Gilbert, 2014), accessed through the NLTK library.
- Implementations of the R_{Flesch} readability metric is provided by the Textstat 0.6.0 Python package[4]. The corpus of easy words for $R_{difficult}$ is also made available by this package.
- Valence, arousal word ratings are obtained from the dataset described in (Warriner et al., 2013)[5]. When computing average valence and arousal for a sentence, stop words are removed and when a word rating cannot be found, a value of 5 is used for valence and 4 for arousal (the mean word ratings).
- Concreteness word ratings are obtained from the dataset described in (Brysbaert et al., 2014) [6]. The concreteness score of a sentence is computed similar to valence and arousal, with a mean word rating of 5 used when no value for a word is available.
- For the n-gram models, a vocabulary size of 1000 was used for all models, and lower-casing was applied for the character and word models.
- The SBERT (Reimers and Gurevych, 2019) implementation and pre-trained models are used for text embedding[7].

[4] Available online here: `https://github.com/shivam5992/textstat`

[5] Available online at `http://crr.ugent.be/archives/1003`.

[6] Available online at `http://crr.ugent.be/archives/1330`.

[7] Can be found online at `https://github.com/UKPLab/sentence-transformers`. We use the `bert-base-nli-mean-tokens` pre-trained model.

- All neural networks using the SBERT embeddings were implemented with the Keras library (Chollet and others, 2015) with the Adam optimizer (Kingma and Ba, 2014) (with default Keras settings) and binary cross-entropy loss. Early stopping is done after validation loss stops decreasing for 4 epochs – with a maximum of 100 epochs. In the deep version of the models, we include two additional densely connected layers as shown in Figure 3, with the second additional layer having half the width of the initial one. We use selu activations (Klambauer et al., 2017) for the additional layers and a dropout rate of 0.5 for only the first additional densely connected layer (Hinton et al., 2012). The hyperparameters requiring tuning for each model and the range of values tested (grid search) is provided in Table A.1.

- The clickbait identification dataset introduced by (Chakraborty et al., 2016) is used, which contains 16,000 clickbait samples and 16,000 non-clickbait headlines[8].

- The headline popularity dataset introduced by (Moniz and Torgo, 2018) is used, which includes feedback metrics for about 100,000 news articles from various social media platforms[9]. For preprocessing, we remove those article where no popularity feedback data is available, and compute popularity by averaging percentiles across platforms. For example, if an article is in the 80^{th} popularity percentile on Facebook and in the 90^{th} percentile on LinkedIn, then it is given a popularity score of 0.85.

- We use the following summarizers: TextRank (Mihalcea and Tarau, 2004), SumBasic (Nenkova and Vanderwende, 2005), LexRank (Erkan and Radev, 2004), and KLSum (Haghighi and Vanderwende, 2009)[10].

- We used the Scikit-learn (Pedregosa et al., 2011) implementations of AdaBoost, decision trees, and logistic regression. To accommodate the imbalanced training data, balanced class weighting was used for the decision trees in Adaboost and logistic regression. For AdaBoost, we use 100 estimators with the default learning rate of 1.0. For logistic regression we use the default settings of L2 penalty with $C = 1.0$.

model	**Initial width**	**# Experts**
A-basic	-	-
A-deep	[16, 32, 64, 128, 256, 512]	-
B-basic	-	-
B-deep	[16, 32, 64, 128, 256, 512]	-
C-basic	-	[2, 4, 8, 16]
C-deep	[16, 32, 64, 128, 256, 512]	[2, 4, 8, 16]

Table A.1: Hyperparameter values used in grid search for the different SBERT neural networks. The models with the best performance on the validation set averaged across 5 trials are reported in Table 3.

Appendix B TF-IDF Analysis of SBERT Embedding Dimensions

In order to uncover the key terms associated with increased PQ probability for a given SBERT embedding dimension, the following steps were performed:

1. Train a logistic regression model using that single feature. Make a note of whether the coefficient is positive (i.e. increasing the feature value increase PQ probability) or negative (i.e. decreasing feature value increases PQ probability).

[8] Available online at `https://github.com/bhargaviparanjape/clickbait/tree/master/dataset`.

[9] Available online at `https://archive.ics.uci.edu/ml/machine-learning-databases/00432/Data/`.

[10] Implementations provided by Sumy library, available at `https://pypi.python.org/pypi/sumy`.

2. Take all test sentences and split them into three groups: (1) those where the feature value is in the top k, (2) those where the feature value is in the middle $2k$, and (3) those where the feature value is in the bottom k. We use $k = 2000$.

3. Join together the sentences within each of the three groups so that we have three "documents" and apply TF-IDF on this set of documents. We use the Scikit-learn (Pedregosa et al., 2011) implementation, with an n-gram range of 1-3 words and use the English stopword list with `sublinear_tf = True`. All other settings are at the default values.

4. If the coefficient from step 1 is positive, use the highest ranked terms for group 1. If the coefficient is negative, use the highest ranked terms for group 3.

Appendix C Model-Chosen Pull Quote Examples

Model	**Highest rated sentence(s)**
True PQ Source	"To date, the fishing industry in British Columbia has not raised the carbon tax as an area of specific concern," it says.
Quote_count	OTTAWA - The federal government's carbon tax could take a toll on Canada's fishing industry, causing its competitiveness to "degrade relative to other nations," according to an analysis from the fisheries department.
Sent_position	In the aquaculture and seafood processing industries, in contrast, fuel makes up just 1.6 per cent and 0.8 per cent of total costs, respectively.
R_difficult	That would result in a difference in the GDP of about $2 billion in 2022, or 0.1 per cent.
POS_PRP	"To date, the fishing industry in British Columbia has not raised the carbon tax as an area of specific concern," it says.
POS_VB	"The relatively rapid introduction of measures to reduce GHG emissions would allow little time for industry and consumers to adjust their behaviour, creating a substantial risk of economic disruption and uncertainty."
A_concreteness	"This could have a negative impact on the competitiveness of Canada's fishing industry."
Char-2	"However, Canada's competitiveness may degrade relative to other nations that have not yet announced plans, or are proceeding more slowly towards measures to reduce GHG emissions," the memo says.
Word-1	The memo concludes that short-term impacts are expected to be "low to moderate," and the department will "continue to monitor developments."
C-deep	"To date, the fishing industry in British Columbia has not raised the carbon tax as an area of specific concern," it says.
Headline popularity	The four largest provinces - Quebec, Ontario, Alberta and B.C.
Clickbait	Ottawa has said all jurisdictions that don't have their own carbon pricing plans in place this year will have the federal carbon tax imposed on them in January 2019, starting at 20$pertonneandincreasingto$50 per tonne in 2022.
TextRank	The analysis was completed in December 2016, shortly after most provinces and territories had signed Ottawa's pan-Canadian climate change framework, committing them to a range of measures, including carbon pricing, to reduce Canada's 2030 emissions to 30 per cent below 2005 levels.

Table C.1: Article source: `https://nationalpost.com/news/politics/federal-carbon-tax-could-degrade-canadian-fishing-industrys-competitiveness-says-memo`.

Model	Highest rated sentence(s)
True PQ Source	I think so many people voted for me because I think they're just proud of me as well.
Quote_count	The school year is finally coming to an end and that means it's prom season, woo season!
Sent_position	I texted my friends like, "Oh my god I'm freaking out.
R_difficult	I'm only at the school for an hour and a half every other day so I had no idea that we were even voting.
POS_PRP	I think so many people voted for me because I think they're just proud of me as well.
POS_VB	- and some people would send me them, but I just choose not to read them.
A_concreteness	I didn't hear about anything.
Char-2	Something that I just want everyone to take away from this is you can be you as long as you're not hurting anyone else and as long as you're not breaking any rules.
Word-1	Something that I just want everyone to take away from this is you can be you as long as you're not hurting anyone else and as long as you're not breaking any rules.
C-deep	I don't think there's any day where I haven't worn a full face of makeup to school, and I always dress up.
Headline popularity	I think so many people voted for me because I think they're just proud of me as well.
Clickbait	I texted my friends like, "Oh my god I'm freaking out.
TextRank	In an interview with Cosmopolitan.com, he talked about putting together his look, why he didn't see his crowning coming, and what he'd like to tell the haters.

Table C.2: Article source: `https://www.cosmopolitan.com/lifestyle/a20107039/south-carolina-prom-king-adam-bell-interview/`

Model	Highest rated sentence(s)
True PQ Source	There is not a downtown in the whole wide world that's made better by vehicle traffic.
Quote_count	We need to stop widening roads and otherwise "improving" our road infrastructure, and pronto.
Sent_position	By putting an immediate moratorium on it.
R_difficult	But at the same time (this is the important part), make it super easy, free (or nearly free) and convenient to get around downtown.
POS_PRP	Not, I think, if we have any say over it.
POS_VB	Have them criss-cross the inner core.
A_concreteness	Not, I think, if we have any say over it.
Char-2	We live far away from where we need to be, and we enjoy activities that aren't always practical by bus, especially if you happen to have kids that need to be in six different places every day.
Word-1	We live far away from where we need to be, and we enjoy activities that aren't always practical by bus, especially if you happen to have kids that need to be in six different places every day.
C-deep	I want to scream.
Headline popularity	Personally, I'd rip out the Queensway and turn it into a light-rail line with huge bike paths, paths for motorcycles, and maybe a lane or two dedicated to autonomous vehicles and taxis and ride-shares.
Clickbait	It's an idea I've been obsessed with since visiting Portland, Oregon, in 2004.
TextRank	Not, I think, if we have any say over it.

Table C.3: Article source: `https://ottawacitizen.com/opinion/columnists/armchair-mayor-fewer-cars-more-transit-options-would-invigorate-ottawa`

Model	Highest rated sentence(s)
True PQ Source	But Pelosi seems to have thought more about alliteration than what pitch would effectively challenge the inaccurate but narratively satisfying story the president had just told.
	Sanders packed more visceral humanity in the first minute or so of his remarks than in the entirety of Pelosi and Schumer's response.
	And perhaps most importantly, he validated that there is, in fact, a crisis afoot: one created by Trump, as well as several produced by structural forces the political class has long ignored.
	And this is an important point: The temptation to fact-check is understandable. And a certain amount of fact-checking is necessary to keep Trump accountable. But poking holes in Trump's narrative, by itself, is not enough.
Quote_count	The life of an American hero was stolen by someone who had no right to be in our country," he said.
Sent_position	An opioid crisis does kill thousands of Americans each year.
R_difficult	The life of an American hero was stolen by someone who had no right to be in our country," he said.
POS_PRP	I'm not going to blame you [Chuck Schumer] for it."
POS_VB	I live paycheck to paycheck, and I can't get a side job because I still have to go to my unpaid federal job."
A_concreteness	He didn't disappoint.
Char-2	"Let me be as clear as I can be," said Sanders, "this shutdown should never have happened."
Word-1	"Let me be as clear as I can be," said Sanders, "this shutdown should never have happened."
C-deep	All are equally guilty - children are merely "pawns," not people.
Headline popularity	And what Trump said about who is hurting most is true: "Among the hardest hit are African-Americans and Hispanic-Americans."
Clickbait	"[Trump] talked about what happened the day after Christmas?
TextRank	These are people in the FBI, in the TSA, in the State Department, in the Treasury Department, and other agencies who have, in some cases, worked for the government for years."

Table C.4: Article source: `https://theintercept.com/2019/01/09/trump-speech-democratic-response/`. This article demonstrates a case where there are many real PQs in an article. It also highlights the need for future work which can create multi-sentence PQs (True PQ #4 consists of two sentences).

Chapter 5

Paper 3

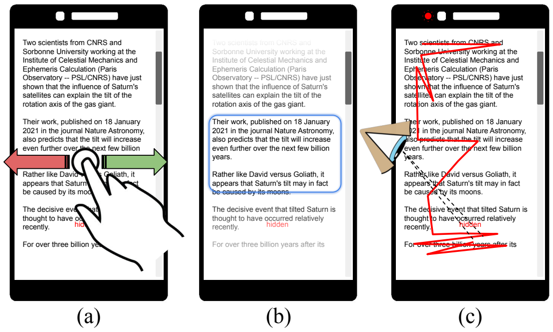
A version of this paper, titled “Hone as You Read: A Practical Type of Interactive Summarization”, was submitted to the Conference of the Association for Computational Linguistics (ACL) 2021 [9]. It is currently under review.

This paper presents a new type of interactive personalized summarization task intended to increase the personal interestingness of an article. This is done by allowing the reader to, in a low-effort and unobtrusive manner, indicate their interest level in sentences as they read. By capturing this feedback as the user reads an article, the article can be fine-tuned to only show content of interest. It is worth mentioning that this new task is a form of informative, rather than indicative, summarization. This is because we assume that the reader is already willing to read (at least some of) the contents of a chosen article, not just get an idea of what *kind* of information is present.

Hone as You Read: A Practical Type of Interactive Summarization

Abstract

We present HARE, a new task where reader feedback is used to optimize document summaries for personal interest *during the normal flow of reading*. This task is related to interactive summarization, where personalized summaries are produced following a long feedback stage where users may read the same sentences many times. However, this process severely interrupts the flow of reading, making it impractical for leisurely reading. We propose to gather minimally-invasive feedback during the reading process to adapt to user interests and augment the document in real-time. Building off of recent advances in unsupervised summarization evaluation, we propose a suitable metric for this task and use it to evaluate a variety of approaches. Our approaches range from simple heuristics to preference-learning and their analysis provides insight into this important task. Human evaluation additionally supports the practicality of HARE. The code to reproduce this work will be made publicly available at `placeholder`.



Two scientists from CNRS and Sorbonne University working at the Institute of Celestial Mechanics and Ephemeris Calculation (Paris Observatory -- PSL/CNRS) have just shown that the influence of Saturn's satellites can explain the tilt of the rotation axis of the gas giant.

Their work, published on 18 January 2021 in the journal Nature Astronomy, also predicts that the tilt will increase even further over the next few billion years.

Rather like David versus Goliath, it appears that Saturn's tilt may in fact ~~be caused by its moons.~~

The decisive event that tilted Saturn is thought to have occurred relatively recently.

For over three billion years after its

Figure 1: Potential feedback methods for HARE used on a smartphone. In (a), users can choose to swipe left or right to indicate interest or disinterest in sections of text as they read. Users may also provide implicit feedback in the form of dwell time in center window (b) or gaze location, as measured by camera for example (c). More interesting text may have longer gazes or dwell time. The approaches evaluated in this paper rely on feedback similar to (a), but further development in HARE can extend to (b) or (c).

1 Introduction

Keeping readers engaged in an article and helping them find desired information are important objectives (Calder et al., 2009; Nenkova and McKeown, 2011). These objectives help readers deal with the explosion of online content and provide an edge to content publishers in a competitive industry. To help readers find personally relevant content while maintaining the flow of natural reading, we propose a new text summarization problem where the summary is **h**oned **a**s you **re**ad (HARE). The challenge is to learn from unobtrusive user feedback, such as the types in Figure 1, to identify uninteresting content to hop over.

This new task is related to both query-based summarization (QS) and interactive personalized summarization (IPS). In QS, users must specify a query to guide the resultant summary (Damova and Koychev, 2010). For users performing focused research, specifying queries is useful, but for more leisurely reading, this requirement interrupts the natural flow. Approaches to IPS avoid the problem of having to explicitly provide a query. However, they suffer a similar problem by requiring users to go through several iterations of summary reading and feedback-providing before a final summary is produced (Yan et al., 2011; Avinesh et al., 2018; Gao et al., 2019; Simpson et al., 2019).

In contrast, HARE places high importance on non-intrusiveness by satisfying multiple properties detailed in Section 3.1 (such as feedback being non-invasive). We find that due to the high cost of generating a dataset for this task, evaluation poses a difficulty. To overcome this, we adapt recent research in unsupervised summary evaluation. We also describe a variety of approaches for HARE that estimate *what* the user is interested in and *how much* they want to read. Automated evaluation finds that relatively simple approaches based on hiding sentences nearby or similar to disliked ones, or explicitly modelling user interests, outperforms the control, where no personalization is done. Human evaluation suggests that not only is deciding the relevance of sentences rather easy in practice, but that even with simple binary feedback, HARE models may truly provide useful reading assistance.

The major contributions of this work are:

1. We define the novel HARE task, and describe a suitable evaluation technique (Section 3).

2. We describe a wide range of motivated approaches for HARE that should serve as useful baselines for future research (Section 4).

3. We evaluate our approaches to gain a deeper understanding of the task (Section 5).

2 Related Work

In this section, we examine related work on QS, IPS, and unsupervised summarization evaluation.

2.1 Query-based Summarization

Both tasks of HARE and QS aim to produce personalized summaries. Unlike generic summarization where many large datasets exist (Hermann et al., 2015; Fabbri et al., 2019; Narayan et al., 2018), development in QS has been affected by a lack of suitable training data (Xu and Lapata, 2020). To cope, approaches have relied on handcrafted features (Conroy et al., 2005), unsupervised techniques (Van Lierde and Chow, 2019), and cross-task knowledge transfer (Xu and Lapata, 2020). The approach of Mohamed and Rajasekaran (2006) highlights how query-based summarizers often work by adapting a generic summarization algorithm and incorporating the query with an additional sentence scoring or filtering component. Alternatively, one can avoid training on QS data by decomposing the task into several steps, each performed by a module constructed for a related task (Xu and Lapata, 2020).

A pervasive assumption in QS is that users have a query for which a *brief* summary is expected. This is reflected in QS datasets where dozens of documents are expected to be summarized in a maximum of 250 words (Dang, 2005; Hoa, 2006) or single documents summarized in a single sentence (Hasselqvist et al., 2017). However, in HARE, we are interested in a wider range of reading preferences. This includes users who are interested in reading the whole article and users whose interests are not efficiently expressed in a written query.

2.2 Interactive Personalized Summarization

The iterative refinement of summaries based on user feedback is also considered by IPS approaches. An early approach by Yan et al. (2011) considers progressively learning user interests by providing a summary (of user-specified length) and allowing them to click on sentences they want to know more about. Based on the words in clicked sentences, a new summary can be generated and the process repeated. Instead of per-sentence feedback, Avinesh and Meyer (2017) allows users to indicate which bigrams of a candidate summary are relevant to their interests. A successor to this system reduces the computation time to produce each summary down to an interactive level of 500ms (Avinesh et al., 2018). The APRIL system (Gao et al., 2019) aims to reduce the cognitive burden of IPS by instead allowing users to indicate preference between candidate summaries. Using this preference information, a summary-ranking model is trained and used to select the next pair of candidate summaries.

Shared among these previous works is that the user is involved in an interactive process which interrupts the normal reading flow with the reviewing of many intermediate summaries. In HARE, the user reads the document as it is being summarized, so that any given sentence is read at most once (if it has not already been removed). These previous works also focus on multi-document summarization, whereas we wish to improve the reading experience during the reading of individual documents.

2.3 Unsupervised Summary Evaluation

When gold-standard human-written summaries are available for a document or question-document pair, the quality of a model-produced summary is commonly computed with the ROUGE metric (Lin and Och, 2004). Driven by high costs of obtaining human-written summaries at a large scale, especially for tasks such as multi-document summarization or QS, unsupervised evaluation of summaries (i.e. without using gold-standards) has rapidly developed (Louis and Nenkova, 2013).

Louis and Nenkova (2009) found that the Jensen Shannon divergence between the word distributions in a summary and reference document outperforms many other candidates and achieves a high correlation with manual summary ratings, but not quite as high as ROUGE combined with reference summaries. Sun and Nenkova (2019) consider a variety of distributed text embeddings and propose to use the cosine similarity of summary and document ELMo embeddings (Peters et al., 2018). Böhm et al. (2019) consider *learning* a reward function from existing human ratings. Their reward function only requires a model summary and document as input and achieves higher correlation with human ratings than other metrics (including ROUGE which requires reference summaries). Stiennon et al. (2020) also consider this approach, with a larger collection of human ratings and larger models. However, Gao et al. (2020) found that comparing ELMo embeddings or using the learned reward from Böhm et al. does not generalize to other summarization tasks. Their evaluation of more advanced contextualized embeddings found that Sentence-BERT (SBERT) embeddings (Reimers and Gurevych, 2019) with word mover's-based distance (Kusner et al., 2015) outperforms other unsupervised options. Post-publication experiments by Böhm et al. further support the generalizability of this approach[1]. In Section 3.3, we adapt the method of Gao et al. to HARE evaluation.

3 Task Formulation

To define the proposed task, we will first describe how a user interacts with an HARE summarizer (Section 3.1). Second, we describe a method for modelling user interests and feedback for automatic evaluation (Section 3.2). Third, we propose an evaluation metric for this new task (Section 3.3).

3.1 User-Summarizer Interaction Loop

The interaction between a user and HARE summarizer, as shown in Figure 2 and sketched in Algorithm 1, consists of the user reading the shown sentences and providing feedback on their relevance. Using this feedback, the summarizer decides which remaining sentences to show, aiming to hide uninteresting sentences. This interaction is designed to smoothly integrate into the natural reading process by exhibiting three important properties: 1) feedback is either implicit or non-intrusive, 2) sentences are presented in their original order to try maintain coherence, and 3) updates to the summary should occur beyond the current reading point so as to not distract the user. Next, we discuss how to model a user in this interaction for the purposes of automatic evaluation.

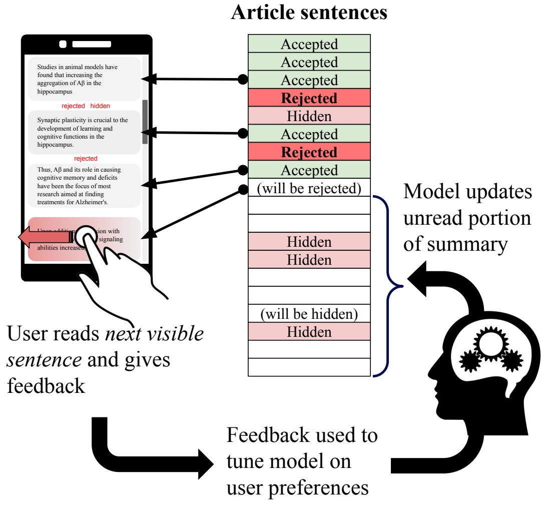

Article sentences
Studies in animal models have found that increasing the aggregation of Aβ in the hippocampus
rejected hidden
Synaptic plasticity is crucial to the development of learning and cognitive functions in the hippocampus.
rejected
Thus, Aβ and its role in causing cognitive memory and deficits have been the focus of most research aimed at finding treatments for Alzheimer's.
sion with
signaling
abilities increased
Accepted
Accepted
Accepted
Rejected
Hidden
Accepted
Rejected
Accepted
(will be rejected)
Hidden
Hidden
(will be hidden)
Hidden
Model updates unread portion of summary
User reads *next visible sentence* and gives feedback
Feedback used to tune model on user preferences

Figure 2: In HARE, users are shown sentences in their original order, and can provide relevance feedback. A model uses this feedback to optimize the remainder of the article, automatically hiding uninteresting sentences.

3.2 User Modelling

In order to model user interaction during HARE, we need to know what kind of feedback they would provide when shown a sentence. This requires understanding how much a user would be interested in a given sentence and how feedback is provided.

User interests For our work, user interests will be modelled as a weighted set of concept vectors from a semantic embedding space. Given a weighted set of k user interests, $U = \{< w_1, c_1 >, ..., < w_k, c_k >\}$ such that $w_i \in [0, 1]$ and $\max(w) = 1$, and a sentence embedding, x, the interest level (which we also refer to as importance) is calculated with Equation 1. We use cosine distance for Δ. Intuitively, the importance of a sentence reflects the maximum weighted similarity to any of the interests. This method of computing importance is similar to that use by Avinesh et al. (2018); Wu et al. (2019); Teevan et al. (2005). However, we adapt it to accommodate modern distributed sentence embeddings (SBERT).

$$R(U, x) = \max_{i=1,...,k} w_i(1 - \Delta(c_i, x)) \qquad (1)$$

[1] The additional results can be found here: https://github.com/yg211/summary-reward-no-reference.

Algorithm 1: User-Summarizer Interaction

```
user chooses a document $D = [x_1, ..., x_{|D|}]$
  to read with help from summarizer $M$
$S = \emptyset$ // summary sentences
for $i$ = 1, ..., $|D|$ do
    if $M$ decides to show $x_i$ to user then
        show sentence $x_i$ to user
        $S := S \cup \{x_i\}$
        incorporate any feedback into $M$
    end
    if user is done reading then
        break
    end
end
return S
```

Feedback types Given a sentence interest score of $r_x \in [0, 1]$, what feedback will be observed by the model? If using implicit feedback like dwell time or gaze tracking, feedback could be continuously valued. With explicit feedback, like ratings or thumbs up/down, feedback could be discrete. For an in-depth discussion on types of user feedback, see Jayarathna and Shipman (2017).

In this work, we will consider an explicit feedback inspired by the "Tinder sort" gesture popularized by the Tinder dating app[2], where users swipe left to indicate disinterest, and right to indicate interest. This feedback interaction has proven to be very quick and easy. Users will routinely sort through hundreds of items in a sitting (David and Cambre, 2016). To adapt this feedback method to our interactive summarization system, we can consider users to "accept" a sentence if they swipe right, and "reject" it if they swipe left (see Figure 1a and Figure 2)[3].

To model the noisy feedback a user provides, we adopt a logistic model, shown in Equation 2, following Gao et al. (2019); Viappiani and Boutilier (2010); Simpson et al. (2019). Our feedback model is parameterized by a decision threshold, $\alpha \in [0, 1]$, and a noise level, $m > 0$. Low α means that users are willing to accept sentences with lower importance. We consider the model to receive a feedback value of 0 if they reject a sentence, and 1 if they accept. In setting α for feedback modelling, we tie it to the users length preference to better simulate realistic behavior. When users want to read very little for example, they only accept the best sentences. If a user wants to read l out of $|D|$, then we set $\alpha = 1 - l/|D|$. For user modelling, we sample l uniformly from the range $[1, |D|]$.

$$P_{\alpha,m}(\text{accept } x) = 1 - \left[1 + exp\left(\frac{\alpha - r_x}{m}\right)\right]^{-1} \tag{2}$$

3.3 Unsupervised Evaluation

Unsupervised evaluation is tricky to do properly. You must show that it correlates well with human judgement, but also be confident that maximizing the metric does not result in garbage (Barratt and Sharma, 2018).

As discussed in Section 2, we adapt the unsupervised summary evaluation method described by Gao et al. (2020). This metric computes a mover's-based distance between the SBERT embeddings of the summary and a heuristically-chosen subset of document sentences (a "pseudo-reference" summary). They show that it correlates well will human ratings and that using it as a reward for training a reinforcement learning-based summarizer produces state-of-the-art models. The authors found that basing the pseudo-reference summary on the lead heuristic, which generally produces good single and multi-document summaries, worked best. For HARE, we can apply the analogous idea: when computing the summary score, we can use all document sentences in the pseudo-reference summary, but weight them by their importance:

$$score(U, D, S) = 1 - \frac{1}{\sum_{x \in D} r_x} \sum_{x \in D} r_x \min_{s \in S} \Delta(x, s) \tag{3}$$

This metric has the behavior of rewarding cases where an important sentence is highly similar to at least one summary sentence. For this reason, coverage of the different user interests is also encouraged by this metric: since sentences drawing their importance from similarity to the same concept are going to be similar to each other, having summaries representing a variety of important concepts is better.

4 Methods

We consider three groups of approaches ranging in complexity: (1) simple heuristics, (2) adapted generic summarizers, and (3) preference learning.

[2] https://tinder.com/?lang=en

[3] If we wanted to make the feedback optional, we could simply let no swipe indicate acceptance, and left swipe indicate rejection.

4.1 Simple Heuristics

This first set of approaches are as follows:

ShowModulo This approach shows every k^{th} sentence to the user. When $k = 1$, this is equivalent to the control, where every sentence is shown. By moving through the article faster, we suspect that greater coverage is obtained, making it more likely that important concepts are represented.

HideNext This approach shows all sentences, except for the k following any rejected sentence. E.g. when $k = 2$ and the user rejects a sentence, the two after it are hidden. The motivation for this model is that nearby sentences are often related, so if one is disliked, a neighbour might also be. Larger k suggests a larger window of relatedness.

HideAllSimilar While HideNext hides physically nearby sentences, this model hides all sentences that are actually conceptually similar to a rejected one, where similarity is measure with cosine similarity of SBERT embeddings. We also include a compromise between hiding based on physical and conceptual similarity: **HideNextSimilar**. This model hides only the unbroken chain of similar sentences after a rejected one.

4.2 Adapted Generic Summarizers

This set of approaches make use of generic extractive summarizers. The motivation for considering them is that even though they are independent of user interests, they are often designed to provide good coverage of an article. In this way, they may accommodate all user interests to some degree. For a given generic summarizer, we consider the following options:

GenFixed This approach first uses the generic summarizer to rank the sentences, and then shows a fixed percentage of the top sentences.

GenDynamic This approach estimates an importance threshold, $\hat{\alpha}$, of sentences the user is willing to read, and hides the less important sentences. Importance is computed by scoring the sentences with the generic summarizer and rescaling the values to $[0, 1]$. The initial estimate is $\hat{\alpha} = 0$, which means that all sentences are important enough. Each time a sentence is rejected, the new estimate is updated to be the average importance of all rejected sentences. To help avoid prematurely extreme estimates, we also incorporate ϵ-greedy exploration. With probability $1 - \epsilon$, the sentence is only shown if the importance meets the threshold, otherwise it is shown anyways. A larger ϵ will help find a closer approximation of the threshold, but at the cost of showing more unimportant sentences.

4.3 Preference Learning

The approaches in this group use more capable adaptive algorithms to learn user preferences in terms of both preferred length and concepts:

LR This approach continually updates a logistic regression classifier to predict feedback given sentence embeddings. Before a classifier can be trained, all sentences are shown. We propose two variations of this approach. The first uses an ϵ-greedy strategy similar to GenDynamic. The second uses an ϵ-decreasing strategy: for a sentence at a given fraction, $frac$, of the way through the article, $\epsilon = (1 - frac)^{\beta}$, for $\beta > 0$.

CoverageOpt This approach explicitly models user interests and length preference. It scores potential sentences by how much they improve coverage of the user interests. However, since we do not know the user's true interests or their length preference, both are estimated as they read.

This approach prepares for each article by using K-Means clustering of sentence embeddings to identify core concepts of the article. The initial estimate of concept importances is computed with:

$$\hat{C} = \left[1 + exp\left(\frac{\mathit{cfsum}}{\beta}\right)\right]^{-1} \tag{4}$$

We initialize the vector *cfsum* with the same value $c \in \mathbb{R}$ for each concept. A larger c means that more evidence is required before a concept is determined to be unimportant. $\beta > 0$ controls how *smoothly* a concept shifts between important and unimportant (larger value means more smoothly). To update the estimate of user interests with *feedback* $\in \{0, 1\}$ for sentence x, we update *cfsum* with:

$$\mathit{cfsum} \leftarrow \mathit{cfsum} + 2(\mathit{feedback} - 0.5)\mathit{concepts}(x) \tag{5}$$

If $feedback = 0$ for example, this moves *cfsum* away from the article concepts represented by that sentence. The function *concepts*() returns the relevance of each concept for the specified sentence.

After updating $\hat{C}$, we re-compute sentence importances based on their contribution to improving concept coverage, weighted by concept importance. Next, we update the estimated length preference,

$\hat{l}_{frac}$, by averaging the importance of rejected sentences. The summary is updated to show sentences among the top $\hat{l}_{frac}|D|$ important sentences. If the user has rejected low and medium importance sentences, then only the most coverage-improving sentences will be shown.

5 Experiments

In this section, we first describe the experimental setup, and then provide an analysis of the results.

5.1 Setup

Dataset We evaluate on the test articles from the non-anonymized CNN/DailyMail dataset (Hermann et al., 2015)[4]. We remove articles with less than 10 sentences so as to cluster sentences into more meaningful groups for user interest modelling. This leaves us with 11222 articles, with an average of 34.0 sentences per article.

User modelling We apply K-Means clustering to SBERT sentence embeddings for each article to identify $k = 4$ cluster centers/concepts. User interests are a random weighting over these concepts, as described in Section 3.2. For feedback noise, we use $m = 0.01$ (essentially no noise) and $m = 0.1$ (intended to capture the difficulty in deciding whether a single sentence is of interest or not). α is chosen as described in Section 3.2.

Metrics Evaluation with the two noise values of $m = 0.01$ and $m = 0.1$ correspond to $score_{sharp}$ and $score_{noisy}$ respectively. $score_{adv}$ corresponds to the difference between $score_{noisy}$ and the control score (no personalization). Positive values indicate outperforming the control. Since the scores fall between 0 and 1, we multiply them by 100.

Privileged information comparison models We consider for comparison three oracle models and the control. OracleGreedy has access to the user preferences and greedily selects sentences to maximize the score, until the length limit is reached. OracleSort selects sentences based only on their interest level. OracleUniform selects sentences at random throughout the article until the length limit is reached[5].

Model	$score_{sharp}$	$score_{noisy}$	$score_{adv}$
OracleGreedy	87.04		4.89
OracleSorted	82.74		0.58
OracleUniform*	82.77		0.62
Control (show all)	82.15		0.0
ShowModulo	78.83		-3.32
HideNext	82.66	82.66	0.51
HideNextSimilar	82.79	82.86	0.71
HideAllSimilar	83.03	**83.09**	**0.94**
GenFixed	81.97		-0.19
GenDynamic*	82.39	82.24	0.09
LR (ϵ-greedy)*	82.48	82.50	0.34
LR (ϵ-decreasing)*	82.28	82.31	0.15
CoverageOpt	**83.11**	82.81	0.65

Table 1: A comparison of each model proposed. For parameterized models, results with the best variation are reported (for all models, we found that the same parameters performed best for both $score_{sharp}$ and $score_{noisy}$). Non-deterministic models are marked by a *. $score_{adv}$ is the difference between $score_{noisy}$ and the control score (which is independent of feedback).

5.2 Results

Table 1 reports the results for each model with its best performing set of hyperparameters. While $score_{sharp}$ and $score_{noisy}$ can range from 0 to 100, the difference between the control and OracleGreedy is less that 5 points (reflected in $score_{adv}$). This suggests that even relatively small performance differences are important. For stochastic models (marked by a * in Table 1), results are averaged across 3 trials and standard deviations were all found to be below 0.05.

Overall, we find that the simple heuristics provide robust performance, unaffected (and possibly helped) by noise. While the more complex CoverageOpt approach is able to perform best with low-noise feedback, it falls behind when noise increases. Next we discuss in more detail the results for each group of models, then comment on aspects of efficiency, and finally discuss the results of our human evaluation.

5.2.1 Privileged Information Models

OracleUniform outperforms the control as well as OracleSorted. This may seem counter-intuitive, since OracleUniform has the disadvantage of not knowing true user interests. However, the strength of OracleUniform is that it provides uniform coverage over the whole article, weakly accommodating any interest distribution. By choosing only the most interesting sentences, OracleSorted runs the risk of only showing those related to the most important concept. If

[4] Accessed through HuggingFace: https://huggingface.co/datasets/cnn_dailymail.

[5] Readers cannot be guaranteed a uniform sampling of sentences unless their length preference is known in advance.

our user model simulated more focused interests, OracleSorted may perform better however.

It is also interesting to see how much higher OracleGreedy is than every other model, suggesting that there is plenty of room for improvement. The reason the oracle does not reach 100 is that the summary length is restricted by user preference. If future approaches consider abstractive summarization techniques, it may be possible to move beyond this performance barrier.

5.2.2 Simple Heuristics

While we suspected that the ShowModulo strategy might benefit from exposing readers to more concepts faster, we found that this does not work as well as OracleUniform. The top performance of $score_{adv} = -3.32$ is reached with $k = 2$, and it quickly drops to -7.06 with $k = 3$. The minimally adaptive approach of hiding a fixed number of sentences after swiped ones, as per HideNext, does help however, especially with $n = 2$.

The related models of HideNextSimilar and HideAllSimilar, which simply hide sentences similar to ones the user swipes away, work surprisingly well, in both moderate and low noise. In Figure 3, we can see that their performance peaks when the similarity threshold is around 0.5 to 0.6.

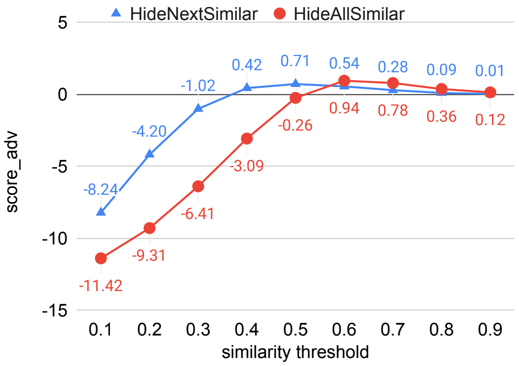


Figure 3: The performance for HideNextSimilar and HideAllSimilar for a range of similarity thresholds. When the threshold is high, it means that only the most similar sentences are hidden.

5.2.3 Adapted Generic Summarizers

We use the following extractive summarizers: LexRank (Erkan and Radev, 2004), SumBasic (Nenkova and Vanderwende, 2005), and TextRank (Mihalcea and Tarau, 2004)[6].

LR (constant ϵ)		LR (decreasing ϵ)	
ϵ	$score_{adv}$	β	$score_{adv}$
0	-7.27	0.25	0.05
0.1	-1.58	0.5	0.09
0.2	-0.18	1	0.15
0.3	0.25	2	0.07
0.4	0.34	4	-0.61
0.5	0.34		

Table 2: Results for the two LR model version. For the constant-ϵ variation, a greater ϵ indicates greater exploration. For the decreasing-ϵ variation, larger β indicates a faster decay in exploration probability.

We find that the generic summarizer-based models always perform worse than the control when showing a fixed fraction of the article (GenFixed). The best model of this type used the SumBasic summarizer, showing 75% of sentences. When dynamically estimating target summary length (GenDynamic), the control is outperformed by only 0.09 points. This is achieved by the SumBasic summarizers with $\epsilon = 0.5$. For both variations, we find that the best hyperparameters are tend to be those that make them show the most sentences.

5.2.4 Preference-learning Models

The LR models out-perform the control, as shown in Table 2, but fail to match the simpler approaches. Using a decaying ϵ actually hurt performance, suggesting that the model is simply not able to learn user preferences fast enough. However, there is a sweet spot for the rate of ϵ decay at $\beta = 1$.

We find that CoverageOpt consistently improves with larger initial concept weights (c) and a slower concept weight-saturation rate (β), with the performance plateauing around $\beta = 4$ and $c = 5$. When both c and β are both large, there is a longer exploration phase with more evidence required to indicate that any given concept should be hidden.

5.3 Efficiency

Acceptance rate When measuring the fraction of shown sentences that are accepted, we find no consistent connection to their performance. For example, the control and the best HideNext, HideNextSimilar, HideAllSimilar, and CoverageOpt models all have rates between 64-66% in the noisy feedback case. OracleSorted has the highest however, at 79%, while OracleGreedy is only at 69% acceptance. As discussed in Section 5.2.1, this is because the sentence set

[6] Implementations provided by Sumy library, available at https://pypi.python.org/pypi/sumy.

which maximizes the score is not necessarily the same as the set with the highest importance sum.

Speed The approaches presented here are able to update the summary in real-time. Running on a consumer-grade laptop, each full user-article simulation (which consists of many interactions) takes between 100ms for the slowest model (GenFixed with TextRank), to 2.8ms for HideAllSimilar, to 1.3ms for HideNext.

5.4 Human Evaluation

Finally, we run a human evaluation to test a variety of approaches on multiple measures.

Setup We selected 10 news articles from a variety of sources and on a variety of topics (such as politics, sports, and science), with an average sentence length of 20.6, and asked 13 volunteers to read articles with the help of randomly assigned HARE models. In total, we collected 70 trials. Participants were shown sentences one at a time and provided feedback to either accept or reject sentences. They were also able to stop reading each article at any time. After reading each article, they were asked several questions about the experience, including the coherence of what they read (how well-connected consecutive sentences were, from 1 to 5) and how easy it was to decide whether to accept or reject sentences (from 1 to 5). We also showed them any unread sentences afterwards in order to determine how many would-be accepted sentences were not shown. Coverage, roughly corresponding to our automated evaluation metric, can then be estimated with the fraction of interesting sentences that were actually shown.

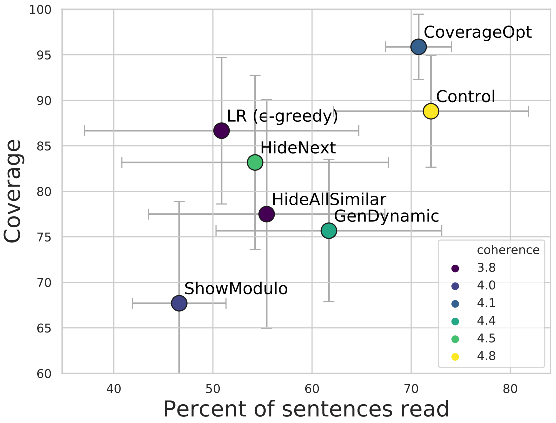


Figure 4: Summary of human evaluation results. Error bars indicate 90% confidence intervals.

Results From the human evaluation, we find that making the decision to accept or reject sentences is quite easy, with an average decision-ease rating of 4.4/5. However, departing from the assumptions of our user model, people ended up reading more than an average of 50% of the articles (up to 70% for the control). This could influence the relative performance of the various models, with a skew towards models that tend to hide fewer sentences. We find the acceptance rate to vary from 47% for LR to 75% for CoverageOpt, with the remainder around 60%. From Figure 4 we can see that the best model (highest coverage) appears to be CoverageOpt. This is followed by the control and LR model, with their 90% confidence intervals overlapping. This highlights that achieving good coverage of interesting sentences is not the same as achieving a high acceptance rate. The worst performing model according to both human and automated evaluation is ShowModulo. The remaining four models significantly overlap in their confidence intervals. However, it is interesting to note that HideAllSimilar performs poorer than we would expect. Given the positive correlation between the percent of the article users end up reading and the model coverage, we can guess that this is a result of the model automatically hiding too many sentences. This also leads to low reported summary coherence, as many sentences are skipped. In contrast, the control achieves the highest coherence (since nothing is skipped), with CoverageOpt near the middle of the pack.

6 Conclusion

In this paper we proposed a new interactive summarization task where the document is automatically refined during the normal flow of reading. By not requiring an explicit query or relying on time-consuming and invasive feedback, relevant information can be conveniently provided for a wide range of user preferences. We provided an approximate user model and suitable evaluation metric for this task, building upon recent advances in unsupervised summary evaluation. To guide examination of this new task, we proposed a variety of approaches, and perform both automated and human evaluation. Future research on this task includes adapting the interaction model to implicit feedback and trying more advanced approaches. We could also consider potential improvements upon the unsupervised evaluation method, possibly by drawing on recent developments in topic modelling (Zhao et al., 2021).

7 Ethical Considerations

Diversity of viewpoints The HARE task is intended for the design of future user-facing applications. By design, these applications have the ability to control what a user reads from a given article. It is possible that, when deployed without sufficient care, these tools could exacerbate the "echo chamber" effect already produced by automated news feeds, search results, and online communities (Pariser, 2011). However, the ability to influence what readers are exposed to can also be leveraged to *mitigate* the echo chamber effect. Rather than considering only what user interests appear to be at a given moment, future HARE models could incorporate a diversity factor to explicitly encourage exposure to alternative views when possible. The weighting of this factor could be tuned to provide both an engaging reading experience and exposure to a diversity of ideas.

Beneficiaries As mentioned in Section 1, those most likely to benefit from HARE applications once successfully deployed will be those using them to read (by saving time and increased engagement) as well as any content publishers who encourage their use.

References

PVS Avinesh, Carsten Binnig, Benjamin Hättasch, Christian M Meyer, and Orkan Özyurt. 2018. Sherlock: A system for interactive summarization of large text collections. *Proc. VLDB Endow.*, 11(12):1902–1905.

PVS Avinesh and Christian M Meyer. 2017. Joint optimization of user-desired content in multi-document summaries by learning from user feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1353–1363.

Shane Barratt and Rishi Sharma. 2018. A note on the inception score. *arXiv preprint arXiv:1801.01973*.

Florian Böhm, Yang Gao, Christian M Meyer, Ori Shapira, Ido Dagan, and Iryna Gurevych. 2019. Better rewards yield better summaries: Learning to summarise without references. *arXiv preprint arXiv:1909.01214*.

Bobby J Calder, Edward C Malthouse, and Ute Schaedel. 2009. An experimental study of the relationship between online engagement and advertising effectiveness. *Journal of interactive marketing*, 23(4):321–331.

John M Conroy, Judith D Schlesinger, and Jade Goldstein Stewart. 2005. Classy query-based multi-document summarization. In *Proceedings of the 2005 Document Understanding Workshop, Boston*. Citeseer.

Mariana Damova and Ivan Koychev. 2010. Query-based summarization: A survey.

Hoa Trang Dang. 2005. Overview of duc 2005. In *Proceedings of the document understanding conference*, volume 2005, pages 1–12.

Gaby David and Carolina Cambre. 2016. Screened intimacies: Tinder and the swipe logic. *Social media+ society*, 2(2):2056305116641976.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Alexander R. Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R. Radev. 2019. Multi-news: a large-scale multi-document summarization dataset and abstractive hierarchical model.

Yang Gao, Christian M Meyer, and Iryna Gurevych. 2019. Preference-based interactive multi-document summarisation. *Information Retrieval Journal*, pages 1–31.

Yang Gao, Wei Zhao, and Steffen Eger. 2020. Supert: Towards new frontiers in unsupervised evaluation metrics for multi-document summarization. *arXiv preprint arXiv:2005.03724*.

Johan Hasselqvist, Niklas Helmertz, and Mikael Kågebäck. 2017. Query-based abstractive summarization using neural networks.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701.

TD Hoa. 2006. Overview of duc 2006. In *Document Understanding Conference*.

Sampath Jayarathna and Frank Shipman. 2017. Analysis and modeling of unified user interest. In *2017 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 298–307. IEEE.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 605–612.

Annie Louis and Ani Nenkova. 2009. Automatically evaluating content selection in summarization without human models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 306–314.

Annie Louis and Ani Nenkova. 2013. Automatically assessing machine summary content without a gold standard. *Computational Linguistics*, 39(2):267–300.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411.

Ahmed A Mohamed and Sanguthevar Rajasekaran. 2006. Improving query-based summarization using document graphs. In *2006 IEEE international symposium on signal processing and information technology*, pages 408–410. IEEE.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *ArXiv*, abs/1808.08745.

Ani Nenkova and Kathleen McKeown. 2011. *Automatic summarization*. Now Publishers Inc.

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

Eli Pariser. 2011. *The filter bubble: What the Internet is hiding from you*. Penguin UK.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Edwin Simpson, Yang Gao, and Iryna Gurevych. 2019. Interactive text ranking with bayesian optimisation: A case study on community qa and summarisation. *arXiv preprint arXiv:1911.10183*.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. *arXiv preprint arXiv:2009.01325*.

Simeng Sun and Ani Nenkova. 2019. The feasibility of embedding based automatic evaluation for single document summarization. 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)*, pages 1216–1221.

Jaime Teevan, Susan T Dumais, and Eric Horvitz. 2005. Personalizing search via automated analysis of interests and activities. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 449–456.

Hadrien Van Lierde and Tommy WS Chow. 2019. Query-oriented text summarization based on hypergraph transversals. *Information Processing & Management*, 56(4):1317–1338.

Paolo Viappiani and Craig Boutilier. 2010. Optimal bayesian recommendation sets and myopically optimal choice query sets. In *Advances in neural information processing systems*, pages 2352–2360.

Chuhan Wu, Fangzhao Wu, Mingxiao An, Tao Qi, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with heterogeneous user behavior. 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)*, pages 4876–4885.

Yumo Xu and Mirella Lapata. 2020. Query focused multi-document summarization with distant supervision. *arXiv preprint arXiv:2004.03027*.

Rui Yan, Jian-Yun Nie, and Xiaoming Li. 2011. Summarize what you are interested in: An optimization framework for interactive personalized summarization. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1342–1351.

He Zhao, Dinh Phung, Viet Huynh, Yuan Jin, Lan Du, and Wray Buntine. 2021. Topic modelling meets deep neural networks: A survey. *arXiv preprint arXiv:2103.00498*.

Hone as You Read: A Practical Type of Interactive Summarization - Supplementary Material

1 Experimental Setup

Computing infrastructure All experiments were performed on a machine with an Intel Core i7-6700HQ CPU with 16G RAM and a GeForce GTX 960M GPU.

Hyperparameter searches For parameterized models, grid searches over the following ranges were performed:

- ShowModulo: $k \in \{2, 3, 4, 5\}$
- HideNext: $n \in \{1, 2, 3, 4\}$
- HideNextSimilar and HideAllSimilar: $threshold \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$
- GenFixed: $frac \in \{0.25, 0.5, 0.75\}$
- GenDynamic: $\epsilon \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$
- LR (constant ϵ): $\epsilon \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$
- LR (decreasing ϵ): $\beta \in \{0.25, 0.5, 1, 2, 4\}$
- CoverageOpt: $\beta \in \{0.25, 0.5, 1, 2, 4\}$ and $c \in \{0, 1, 2, 3, 4\}$

2 Detailed Results

Detailed results for those models without full results reported in the paper are shown here. For ShowModulo and HideNext, results are shown in Table 1. For summarizer-based models, results are shown in Table 2. For CoverageOpt, results are shown in Table 3.

ShowModulo		HideNext	
k	$score_{adv}$	n	$score_{adv}$
2	-3.32	1	0.45
3	-7.06	2	0.51
4	-9.87	3	0.19
5	-12.00	4	-0.41

Table 1: Results for the first two simple heuristic models. For ShowModulo, every k^{th} sentence is shown. For HideNext, the n sentences following a swiped one are hidden.

	frac (for GenFixed)					
summarizer	0.25	0.5	0.75			
LexRank	-11.18	-3.77	-0.79			
SumBasic	-10.75	-3.22	-0.19			
TextRank	-12.28	-4.99	-1.53			
	ϵ (for GenDynamic)					
summarizer	0	0.1	0.2	0.3	0.4	0.5
LexRank	-1.37	-0.53	-0.22	-0.07	0.01	0.06
SumBasic	-3.19	-1.47	-0.72	-0.28	-0.05	0.09
TextRank	-1.95	-1.02	-0.59	-0.31	-0.18	-0.08

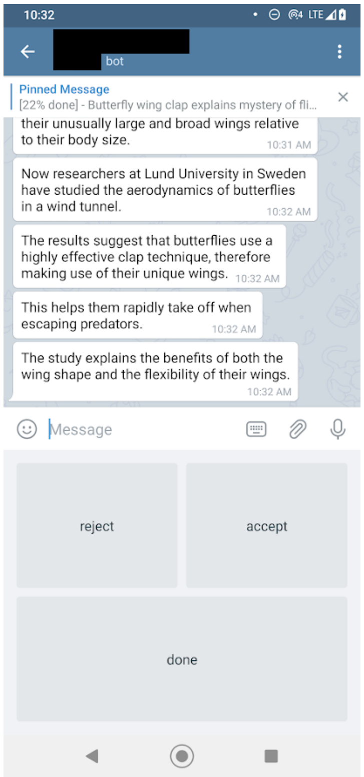
Table 2: Results for the two variations of adapted generic summarizer models, for each of three extractive summarizers tested. For GenFixed, $frac$ indicates what fraction of the document is shown, after first sorting sentences by importance. For GenDynamic, ϵ is used for ϵ-greedy exploration to estimate length preference.

	β				
c	**1/4**	**1/2**	**1**	**2**	**4**
0	0.12	0.22	0.33	0.42	0.50
1	0.51	0.50	0.51	0.52	0.55
2	0.49	0.57	0.60	0.59	0.59
3	0.50	0.53	0.61	0.63	0.63
4	0.49	0.50	0.59	0.64	0.64
5	0.49	0.50	0.55	0.64	**0.65**

Table 3: Results for the COVERAGEOPT model. c controls the initial estimate for concept importances and β controls how smoothly a concept shifts between important and unimportant.

3 Human Evaluation

Human evaluation was performed via a chatbot deployed on the Telegram chat app[1] using their convenient API[2]. A screenshot of the chatbot serving as a simple HARE interface is shown in Figure 1. To participate, volunteers were instructed to engage with the publicly accessible bot in the app and follow instructions provided therein.



[22% done] - Butterfly wing clap explains mystery of fli...

their unusually large and broad wings relative to their body size. 10:31 AM

Now researchers at Lund University in Sweden have studied the aerodynamics of butterflies in a wind tunnel. 10:32 AM

The results suggest that butterflies use a highly effective clap technique, therefore making use of their unique wings. 10:32 AM

This helps them rapidly take off when escaping predators. 10:32 AM

The study explains the benefits of both the wing shape and the flexibility of their wings. 10:32 AM

Figure 1: A screenshot of the demo in action. For each sentence, users were able to accept, reject, or stop reading the article at that point.

[1] https://telegram.org/

[2] https://core.telegram.org/bots/api

Chapter 6

Conclusion

Toward the goal of improving the reading experience through supporting internal reader motivation, we have made contributions to three problems in NLP: coherence modelling (Chapter 3), PQ selection (Chapter 4), and personalized summarization (Chapter 5). In this last chapter we will review our proposed methods for supporting internal motivation (Section 6.1), summarize our findings (Section 6.2), and finally contemplate future research directions (Section 6.3).

6.1 Synopsis

This thesis has studied the problem of supporting internal reader motivation from three directions:

Reading difficulty Chapter 3 presented a new approach for estimating the coherence of a document, a property which affects reading difficulty. In particular, it proposed a new sentence embedding which captures the expected positional distributions of sentences. We evaluated our method on two types of coherence modelling tasks as well as a related task of summarization.

Situational interest Chapter 4 introduced the new task of PQ selection. PQs are elements of a news article specifically designed to catch the attention of readers. We proposed a diverse set of approaches for this task, ranging in complexity. We performed automatic evaluation of these approaches with a newly constructed PQ dataset, as well as human evaluation. An in-depth analysis of several approaches was also performed to gain a deeper understanding of what makes for a good PQ.

Personal interest Chapter 5 introduced a new personalized summarization task called HARE. The goal of this task is to efficiently incorporate user feedback while they read, so as to summarize the remaining unread part of the document to match their personal interests. We proposed a new unsupervised evaluation method which can make use of existing news article datasets, proposed a variety of approaches designed to test several hypotheses about the nature of the new task, and performed both automatic and human evaluation of these approaches.

6.2 Summary of Findings

This thesis followed an explorational aim: to investigate novel approaches to supporting internal reading motivation using machine learning. We now discuss how our work addressed the objectives initially laid out to achieve this aim while providing a summary of our findings.

Identifying existing or new tasks in NLP whose solutions can be used to address reader motivation In Section 1.1.2, we identified an abundance of NLP tasks that can be used to address several aspects of the reading experience (visualized in Figure 1.1). We chose to focus on three tasks where the literature supported their effectiveness at supporting internal motivation in particular. The first task was coherence modelling. Coherence affects the ease of comprehension of a text, with more easily readable texts leading to greater reader enjoyment and interest. The second task was PQ selection. PQs, which often occur in newspapers or online articles, increase the situational interestingness of the article. The third task was HARE, a type of personalized summarization. This task explicitly aims to increase the personal interestingness (i.e., relevance) of the text. While coherence modelling was a pre-existing task, both PQ selection and HARE are new tasks developed during the course of this PhD.

Devising novel machine learning approaches to these tasks For the task of coherence modelling, we introduced PPDs, a sentence embedding learned through self-supervision which reflects the predicted position distribution of a sentence in a document. These embeddings were created following the intuition that if a sentence does not occur where its semantics or style suggest, then it contributes to a lack of coherence. Using these embeddings, we can suggest a coherent ordering of sentences by sorting them based on the weighted average predicted quantiles. To estimate overall coherence, we can compute a correlation coefficient between the suggested ordering and the true ordering.

For the task of PQ selection, we proposed several machine learning approaches. We considered three motivated sets of handcrafted features, including surface-level features, part-of-speech features, and affect features. We considered n-gram features at the character and word level. We also used cross-task models: models trained on tasks we suspected were related to PQ selection. We also considered three groups of neural architectures for the task, including a neural mixture-of-experts.

For HARE, we proposed simple heuristic approaches based on sentence similarity and proximity, cross-task approaches based on generic summarization, and more flexible machine learning approaches to adapt to user feedback. The first machine learning model used logistic regression and distributed sentence embeddings to predict future user feedback. The second machine learning approach was model-based, in that it assumed a model of user behaviour based on a discrete set of user interests and length preference, and used incoming feedback to refine the parameters of that model.

Evaluating and examining these approaches to determine what they can teach us about the various factors of reader motivation For coherence modelling, we performed automatic evaluation on two types of coherence tasks. We demonstrated that the simple PPD-based approach is competitive with more advanced and specialized systems. We saw that this approach is amenable to visual approximation – by simply visualizing the PPD sequence, we can identify problem areas of the text or even identify natural locations to split the text into smaller coherent sections or paragraphs. The quantitative success of this approach supports our intuition on the importance of sentence ordering on coherence and thus readability, based on a normative understanding: a text is coherent if the sentences occur in similar global positions to similar sentences in similar types of documents. In addition, driven by the knowledge that introductory sentences tend to make for good summary sentences, we showed that using PPDs to identify introductory-like sentences outperforms other heuristic extractive summarizers.

We evaluated our PQ selection approaches on a newly constructed dataset for the task. Overall, we found that the neural network approaches performed the best, particularly the mixture-of-experts. The character n-gram model followed. The handcrafted features gave a wide range of performances, with the single best feature being the number of quotation marks present. Among the cross-task models, the clickbait model performed best, supporting the idea that writing a PQ is more about catching reader attention than conveying the important details of the article. A couple more factors that turned out to be important for creating PQs (and thus, catching attention) were the use of abstract subjects, usage of personal pronouns and verbs, and ensuring high readability.Human evaluation of several of the approaches confirmed their general performance ranking, and suggested that the best human-rated model, the character bigram model, performs on par, or even better than copy editor-selected PQs.

For our HARE summarization models, we performed automatic unsupervised evaluation, which relied on an existing dataset of news articles and a model of user interests and behavior. This evaluation was performed in both low and moderate noise settings. Overall, we found that the simple heuristic models performed the best, especially in the moderate-noise setting. In particular, the summarizer which simply hid all sentences semantically similar to disliked sentences performed best. The summarizer which relied on a more detailed model of user behavior in order to optimize coverage of interesting concepts performed best in the low-noise scenario, as well as in the human evaluation. As a result of the human evaluation, we also found that the process of providing simple feedback while reading was rated as very easy. This project supports the idea that improving personal interestingness of reading material does not need to be a time consuming or interruptive process. Rather, this work suggests that enough preference information can be conveyed while reading in order to dynamically update a document to match estimated user interests and length preference.

6.3 Future Directions

While the individual integrated articles briefly mention directions for future work, we recount and expand upon them in this section.

PPDs In Chapter 4, we applied PPDs to estimating coherence in a small number of domains. It may prove insightful to apply it to a wider range of texts, such as conversations or even movie scripts. In this way, we might be able to visually analyze their coherence. Additionally, our evaluation relied on a rather synthetic form of incoherence, namely sentence shuffling. It would be interesting to see how well this approach matches human coherence rating of natural texts.

Integrating PPDs into other NLP tasks such as text generation or author identification may also be considered. Methods for text generation are known to suffer from large-scale incoherence [111], and such a technique may be integrated in a differentiable way to improve coherence of generative methods. For author identification, patterns in the progression of thoughts may form a "fingerprint" which can be captured by PPDs.

PQs In Chapter 4, when constructing our dataset and evaluation methodology, we assumed that all PQs are of the same quality. However, we can presume that some PQs will do a better job at catching attention than others. Discovering what distinguishes between good and great PQs would be interesting. Online news sites may be able to generate this kind of quality data with A/B testing, similar to how many sites already implement such testing for headline optimization [43]. Additionally, our work on PQs focused on *selection*, which captures only some of the true complexity of creating PQs. Rather than only select individual sentences, a PQ generation tool should be able to select one or more neighboring (not necessarily contiguous) sentences and perform augmentations (such as adding details or removing clauses) to turn it into a high-quality PQ. This is also related to the problem of how to suggest larger scale edits to a phrase which can make it more PQ-worthy while maintaining its original meaning and veracity.

In addition to simply selecting or generating PQs, we could consider the problem of recognizing *where* in the text PQs would bring the most value. Perhaps this could be done by estimating where the least interesting parts of the text are? In this way, PQs could maintain reader interest where it is most critical. We can also consider dynamically adding PQs based on real-time estimations of reader engagement or reader interest. If we know that a reader finds some topics more interesting, PQs may be personalized and more effective at maintaining attention.

HARE In Chapter 5, our experiments considered a simple type of explicit feedback for the interactive personalized summarization task. To further reduce the effort required by users to obtain personalized summaries, we should consider implicit feedback methods such as dwell time or gaze tracking. This direction brings extra difficulties however, especially in increased noise of feedback signals. To handle this increased noise, as well as improve other approaches to HARE, we can consider maintaining user profiles, so that we do not need to learn their interests from scratch every time they read a new article. By incorporating profiles, we can also improve the quality of personalized summaries by estimating what the user already knows. In this way, depending on what articles a user has previously read, we could hide more or less of the content in the current article.

Our work on HARE also considered rather simple approaches, which is important to establish baselines for future work. More advanced approaches should certainly be considered in future work. One promising direction is meta-learning, i.e., learning *how to learn* from user feedback for summary personalization. For example, we considered a logistic regression model which was trained with user feedback to predict the feedback on later sentences. With noisy feedback, few training samples, and high-dimensional sentence embeddings, the model is slow to learn. If we could learn in advance what embedding dimensions tend to be the most discriminative or learn what kind of dynamic exploration vs. exploitation strategy to use when deciding to show sentences, performance may be improved.

Beyond text This thesis considered ways of supporting internal motivation when reading. A primary application area of this research is in education, where developing reading skills and obtaining knowledge through reading is important. However, an increasingly large portion of learning these days is done through video [24], especially during the current COVID-19 pandemic where many students are learning from home. If we ask the question of how we can improve the *video watching* experience, we can first see what methods considered in this thesis have video analogues. For example, does the coherence of a video presentation affect viewer enjoyment or understanding of the material being presented? And how might we handle the more complex forms of incoherence in videos which can occur in both the visual and audio dimensions? Does there exist an analogue to the pull quote in video presentations? And can we learn to automatically identify, extract, and leverage these to maintain reader interest? Finally, would it be feasible to adapt the idea of HARE summarizers to videos? That is, can we use minimally invasive viewer feedback to augment a video being watched in real time? Rather than sitting through an hour long lecture and falling asleep, such a personalized video summarizer might identify what parts the student could skip given their interests, current knowledge, or time constraints.

Bibliography

[1] Giuseppe Amato, Fabio Carrara, Fabrizio Falchi, Claudio Gennaro, and Gabriele Lagani. Hebbian learning meets deep convolutional neural networks. In *International Conference on Image Analysis and Processing*, pages 324–334. Springer, 2019.

[2] Avi Assor, Haya Kaplan, and Guy Roth. Choice is good, but relevance is excellent: Autonomy-enhancing and suppressing teacher behaviours predicting students' engagement in schoolwork. *British journal of educational psychology*, 72(2):261–278, 2002.

[3] Regina Barzilay and Noemie Elhadad. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 2002.

[4] Regina Barzilay and Mirella Lapata. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34, 2008.

[5] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

[6] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[7] Tanner Bohn, Yining Hu, Jinhang Zhang, and Charles Ling. Learning sentence embeddings for coherence modelling and beyond. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 151–160, Varna, Bulgaria, September 2019. INCOMA Ltd.

[8] Tanner Bohn and Charles Ling. Catching attention with automatic pull quote selection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 62–76, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[9] Tanner Bohn and Charles Ling. Hone as you read: A practical type of interactive summarization. 2021. Submitted to the Conference of the Association for Computational Linguistics.

[10] Zvia Breznitz and Lauren Berman. The underlying factors of word reading rate. *Educational Psychology Review*, 15(3):247–265, 2003.

[11] Abhijnan Chakraborty, Saptarshi Ghosh, Niloy Ganguly, and Krishna P Gummadi. Optimizing the recency-relevance-diversity trade-offs in non-personalized news recommendations. *Information Retrieval Journal*, 22(5):447–475, 2019.

[12] Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. Stop clickbait: Detecting and preventing clickbaits in online news media. In *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*, pages 9–16. IEEE, 2016.

[13] Yimin Chen, Niall J Conroy, and Victoria L Rubin. Misleading online content: recognizing clickbait as" false news". In *Proceedings of the 2015 ACM on workshop on multimodal deception detection*, pages 15–19, 2015.

[14] François Chollet et al. Keras. `https://keras.io`, 2015.

[15] JW Click and Guido Hermann Stempel. *Reader Response to Modern and Traditional Front Page Make-Up*. American Newspaper Publishers Association, 1974.

[16] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *CoRR*, abs/1705.02364, 2017.

[17] Mariana Damova and Ivan Koychev. Query-based summarization: A survey. 2010.

[18] David Ingram and the Peter Henshall Estate. Journalism & media glossary. `https://www.thenewsmanual.net/Resources/glossary.html`, 2019. Online - Accessed 2021-02-11.

[19] Heidar Davoudi, Aijun An, and Gordon Edall. Content-based dwell time engagement prediction model for news articles. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 226–233, 2019.

[20] Ghislaine Dehaene-Lambertz, Karla Monzalvo, and Stanislas Dehaene. The emergence of the visual word form: Longitudinal evolution of category-specific ventral visual areas during reading acquisition. *PLoS biology*, 16(3):e2004103, 2018.

[21] Department for Education Education Standards Research Team and others. Research evidence on reading for pleasure. `https://www.gov.uk/government/publications/research-evidence-on-reading-for-pleasure`, 2012.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[23] Sagnik Dhar, Vicente Ordonez, and Tamara L Berg. High level describable attributes for predicting aesthetics and interestingness. In *CVPR 2011*, pages 1657–1664. IEEE, 2011.

[24] Prajakta Diwanji, Bindu Puthur Simon, Michael Märki, Safak Korkut, and Rolf Dornberger. Success factors of online learning videos. In *2014 International Conference on Interactive Mobile Communication Technologies and Learning (IMCL2014)*, pages 125–132. IEEE, 2014.

[25] William H DuBay. The principles of readability. *Online Submission*, 2004.

[26] Erick Elejalde, Leo Ferres, and Rossano Schifanella. Understanding news outlets' audience-targeting patterns. *EPJ Data Science*, 8(1):16, 2019.

[27] Brita Elvevåg, Peter W Foltz, Daniel R Weinberger, and Terry E Goldberg. Quantifying incoherence in speech: an automated methodology and novel application to schizophrenia. *Schizophrenia research*, 93(1-3):304–316, 2007.

[28] Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479, 2004.

[29] Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. Summeval: Re-evaluating summarization evaluation. *arXiv preprint arXiv:2007.12626*, 2020.

[30] Youmna Farag, Helen Yannakoudakis, and Ted Briscoe. Neural automated essay scoring and coherence modeling for adversarially crafted input. *arXiv preprint arXiv:1804.06898*, 2018.

[31] Steven R Fischer. *A history of reading*. Reaktion books, 2004.

[32] Terri Flowerday, Gregory Schraw, and Joseph Stevens. The role of choice and interest in reader engagement. *The Journal of Experimental Education*, 72(2):93–114, 2004.

[33] Nigel French. *InDesign Type: Professional Typography with Adobe InDesign*. Adobe Press, 2018.

[34] Edgar Galván and Peter Mooney. Neuroevolution in deep neural networks: Current trends and future challenges. *arXiv preprint arXiv:2006.05415*, 2020.

[35] Yang Gao, Christian M Meyer, and Iryna Gurevych. Preference-based interactive multi-document summarisation. *Information Retrieval Journal*, pages 1–31, 2019.

[36] Nikhil Garg, Benoit Favre, Korbinian Reidhammer, and Dilek Hakkani-Tür. Clusterrank: a graph based method for meeting summarization. In *Tenth annual conference of the international speech communication association*, 2009.

[37] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[38] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[39] Alex Graves. *Supervised sequence labelling with recurrent neural networks*. Studies in computational intelligence. Springer, 2012.

[40] Dan Gunter, Brian L Tierney, Aaron Brown, Martin Swany, John Bresnahan, and Jennifer M Schopf. Log summarization and anomaly detection for troubleshooting distributed systems. In *2007 8th IEEE/ACM International Conference on Grid Computing*, pages 226–234. IEEE, 2007.

[41] John T Guthrie, Angela McRae, and Susan Lutz Klauda. Contributions of concept-oriented reading instruction to knowledge about interventions for motivations in reading. *Educational Psychologist*, 42(4):237–250, 2007.

[42] Yaakov HaCohen-Kerner, Daniel Miller, and Yair Yigal. The influence of preprocessing on text classification using a bag-of-words representation. *PloS one*, 15(5):e0232525, 2020.

[43] Nick Hagar and Nicholas Diakopoulos. Optimizing content with a/b headline testing: Changing newsroom practices. *Media and Communication*, 7(1):117–127, 2019.

[44] Aria Haghighi and Lucy Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370, 2009.

[45] Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

[46] Matthew Hindman. Stickier news. *URL: https://shorensteincenter. org/wp-content/uploads/2015/04/Stickier-News-Matthew-Hindman.pdf*, 2015.

[47] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

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

[49] Tim Holmes. *Subediting and Production for Journalists: Print, Digital & Social*. Routledge, 2015.

[50] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[51] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[52] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[53] Eithne Kennedy, Elizabeth Dunphy, Bernadette Dwyer, Geraldine Hayes, Thérèse McPhillips, Jackie Marsh, Maura O'Connor, and Gerry Shiel. *Literacy in Early Childhood and Primary Education (3-8 Years).* National Council for Cirriculum and Assessment, 2012.

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

[55] Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in Neural Information Processing Systems*, pages 3294–3302, 2015.

[56] Shaohui Kuang, Deyi Xiong, Weihua Luo, and Guodong Zhou. Modeling coherence for neural machine translation with dynamic and topic caches. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 596–606, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

[57] Dmitry Lagun and Mounia Lalmas. Understanding and measuring user engagement and attention in online news reading. In *Proceedings of the ACM International Conference on Web Search and Data Mining*, pages 113–122, 2016.

[58] Mirella Lapata and Regina Barzilay. Automatic evaluation of text coherence: Models and representations. In *IJCAI*, volume 5, pages 1085–1090, 2005.

[59] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014.

[60] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[61] Yann LeCun, Koray Kavukcuoglu, and Clément Farabet. Convolutional networks and applications in vision. In *Proceedings of 2010 IEEE international symposium on circuits and systems*, pages 253–256. IEEE, 2010.

[62] Junghyuk Lee and Jong-Seok Lee. Music popularity: Metrics, characteristics, and audio-based prediction. *IEEE Transactions on Multimedia*, 20(11):3173–3182, 2018.

[63] Haoran Li, Junnan Zhu, Jiajun Zhang, and Chengqing Zong. Ensure the correctness of the summary: Incorporate entailment knowledge into abstractive sentence summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1430–1441, 2018.

[64] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *arXiv preprint arXiv:2006.08218*, 1(2), 2020.

[65] Lajanugen Logeswaran, Honglak Lee, and Dragomir Radev. Sentence ordering and coherence modeling using recurrent neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[66] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165, 1958.

[67] Yu-Fei Ma, Lie Lu, Hong-Jiang Zhang, and Mingjing Li. A user attention model for video summarization. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 533–542, 2002.

[68] MasterClass. How to write a lede in journalism. `https://www.masterclass.com/articles/how-to-write-a-lede-in-journalism#the-history-of-the-lede-lede-vs-lead`, 2020. Online - Accessed 2021-02-11.

[69] Danielle S McNamara. *Reading comprehension strategies: Theories, interventions, and technologies*. Psychology Press, 2007.

[70] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, 2004.

[71] Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, et al. Evolving deep neural networks. In *Artificial intelligence in the age of neural networks and brain computing*, pages 293–312. Elsevier, 2019.

[72] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[73] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[74] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751, 2013.

[75] Tom M Mitchell et al. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, 45(37):870–877, 1997.

[76] Nuno Moniz and Luís Torgo. Multi-source social feedback of online news feeds. *arXiv preprint arXiv:1801.07055*, 2018.

[77] Han Cheol Moon, Tasnim Mohiuddin, Shafiq Joty, and Xu Chi. A unified neural coherence model. *arXiv preprint arXiv:1909.00349*, 2019.

[78] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[79] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

[80] Ramesh Nallapati, Bowen Zhou, and Mingbo Ma. Classify or select: Neural architectures for extractive document summarization. *arXiv preprint arXiv:1611.04244*, 2016.

[81] Ani Nenkova and Kathleen McKeown. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76. Springer, 2012.

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

[83] Ani Nenkova and Lucy Vanderwende. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005*, 101, 2005.

[84] Ethan Pancer, Vincent Chandler, Maxwell Poole, and Theodore J Noseworthy. How readability shapes social media engagement. *Journal of Consumer Psychology*, 29(2):262–270, 2019.

[85] Maxime Peyrard. A simple theoretical model of importance for summarization. *arXiv preprint arXiv:1801.08991*, 2018.

[86] Emily Pitler and Ani Nenkova. Revisiting readability: A unified framework for predicting text quality. In *Proceedings of the 2008 conference on empirical methods in natural language processing*, pages 186–195, 2008.

[87] Martin Potthast, Sebastian Köpsel, Benno Stein, and Matthias Hagen. Clickbait detection. In *European Conference on Information Retrieval*, pages 810–817. Springer, 2016.

[88] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[89] Soheil Rayatdoost and Mohammad Soleymani. Ranking images and videos on visual interestingness by visual sentiment features. In *MediaEval*, 2016.

[90] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[91] Katharina Reinecke, Tom Yeh, Luke Miratrix, Rahmatri Mardiko, Yuechen Zhao, Jenny Liu, and Krzysztof Z Gajos. Predicting users' first impressions of website aesthetics with a quantification of perceived visual complexity and colorfulness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2049–2058. ACM, 2013.

[92] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[93] Frank Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, Cornell Aeronautical Lab Inc Buffalo NY, 1961.

[94] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[95] Sunil Saxena. *Headline writing*. Sage, 2006.

[96] Denise Schmandt-Besserat. The evolution of writing. *Austin, Texas: University of*, 2014.

[97] Gregory Schraw, Roger Bruning, and Carla Svoboda. Sources of situational interest. *Journal of reading behavior*, 27(1):1–17, 1995.

[98] Harald T Schupp, Jessica Stockburger, Maurizio Codispoti, Markus Junghöfer, Almut I Weike, and Alfons O Hamm. Selective visual attention to emotion. *Journal of Neuroscience*, 27(5):1082–1089, 2007.

[99] Liang Shi, Jinqiao Wang, Lei Xu, Hanqing Lu, and Changsheng Xu. Context saliency based image summarization. In *2009 IEEE International Conference on Multimedia and Expo*, pages 270–273. IEEE, 2009.

[100] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014.

[101] Catherine Snow. *Reading for understanding: Toward an R&D program in reading comprehension*. Rand Corporation, 2002.

[102] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[103] Gerald C Stone. *Examining Newspapers: What Research Reveals About America's Newspapers*, volume 20. Sage Publications, Inc, 1987.

[104] James Glen Stovall. *Infographics: A Journalist's Guide*. Allyn & Bacon, 1997.

[105] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[106] Ke Tran, Arianna Bisazza, and Christof Monz. The importance of being recurrent for modeling hierarchical structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4731–4736, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[107] Sandra H Utt and Steve Pasternack. Use of graphic devices in a competitive situation: a case study of 10 cities. *Newspaper Research Journal*, 7(1):7–16, 1985.

[108] Maarten Vansteenkiste, Willy Lens, and Edward L Deci. Intrinsic versus extrinsic goal contents in self-determination theory: Another look at the quality of academic motivation. *Educational psychologist*, 41(1):19–31, 2006.

[109] Wayne Wanta and Dandan Gao. Young readers and the newspaper: Information recall and perceived enjoyment, readability, and attractiveness. *Journalism Quarterly*, 71(4):926–936, 1994.

[110] Wayne Wanta and Jay Remy. Information recall of four newspaper elements among young readers. 1994.

[111] Sam Wiseman, Stuart Shieber, and Alexander Rush. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[112] Hao Xu, Jingdong Wang, Xian-Sheng Hua, and Shipeng Li. Hybrid image summarization. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1217–1220, 2011.

[113] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005.

[114] Rui Yan, Jian-Yun Nie, and Xiaoming Li. Summarize what you are interested in: An optimization framework for interactive personalized summarization. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1342–1351, 2011.

[115] Xin Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

[116] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*, 2019.

[117] Renxian Zhang. Sentence ordering driven by local and global coherence for summary generation. In *Proceedings of the ACL 2011 Student Session*, pages 6–11, 2011.

[118] Richard Zhang. Making convolutional networks shift-invariant again. In *International Conference on Machine Learning*, pages 7324–7334. PMLR, 2019.

[119] Xueyuan Zhou and Mikhail Belkin. Semi-supervised learning. In *Academic Press Library in Signal Processing*, volume 1, pages 1239–1269. Elsevier, 2014.

[120] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-CALD-02-107. 2002.

Curriculum Vitae

Name: Tanner Bohn

Post-Secondary Education and Degrees:

University of Saskatchewan
Saskatoon, Saskatchewan
2011 - 2015 B.Sc.

Western University
London, ON
2015 - 2017 M.SC.

Western University
London, ON
2017 - 2021 Ph.D..

Related Work Experience:

Teaching Assistant
Western University
2015 - 2021

Student Researcher
HighStreet
2016

Student Researcher
Arcane
2016

Undergraduate Research Student, Department of Computer Science
University of Saskatchewan
2013 - 2014

Honours and Awards: Best Student Paper Award at Canadian AI Conference
2020

Best Presentation award in UWORCS
2019

Refereed Conference Publications:

Bohn, T., & Ling, C. (2020). Catching Attention with Automatic Pull Quote Selection. Proceedings of the 28th International Conference on Computational Linguistics. https://doi.org/10.18653/v1/2020.coling-main.6

Yun, X., **Bohn, T.**, & Ling, C. (2020, May). A Deeper Look at Bongard Problems. In Canadian Conference on Artificial Intelligence (pp. 528-539). Springer, Cham.

Bohn, T., Hu, Y., Zhang, J., & X. Ling, C. (2019, October 22). Learning Sentence Embeddings for Coherence Modelling and Beyond. Proceedings - Natural Language Processing in a Deep Learning World. Recent Advances in Natural Language Processing. https://doi.org/10.26615/978-954-452-056-4_018

Journal Papers:

Green, K. R., **Bohn, T. A.**, & Spiteri, R. J. (2019). Direct Function Evaluation versus Lookup Tables: When to Use Which?. SIAM Journal on Scientific Computing, 41(3), C194-C218.