Content Selection in Deep Learning Models of Summarization

Chris Kedzie and **Kathleen McKeown**
Department of Computer Science
Columbia University
{kedzie,kathy}@cs.columbia.edu

Hal Daumé III
University of Maryland, College Park
Microsoft Research, New York City
hal@cs.umd.edu

Abstract

We carry out experiments with deep learning models of summarization across the domains of news, personal stories, meetings, and medical articles in order to understand how content selection is performed. We find that many sophisticated features of state of the art extractive summarizers do not improve performance over simpler models. These results suggest that it is easier to create a summarizer for a new domain than previous work suggests and bring into question the benefit of deep learning models for summarization for those domains that do have massive datasets (i.e., news). At the same time, they suggest important questions for new research in summarization; namely, new forms of sentence representations or external knowledge sources are needed that are better suited to the sumarization task.

1 Introduction

Content selection is a central component in many natural language generation tasks, where, given a generation goal, the system must determine which information should be expressed in the output text (Gatt and Krahmer, 2018). In summarization, content selection is usually accomplished through sentence (and, occasionally, phrase) extraction. Despite being a key component of both extractive and abstractive summarization systems, it is is not well understood how deep learning models perform content selection with only word and sentence embedding based features as input. Non-neural network approaches often use frequency and information theoretic measures as proxies for content salience (Hong and Nenkova, 2014), but these are not explicitly used in most neural network summarization systems.

In this paper, we seek to better understand how deep learning models of summarization perform content selection across multiple domains (§ 4): news, personal stories, meetings, and medical articles (for which we collect a new corpus).[1] We analyze several recent sentence extractive neural network architectures, specifically considering the design choices for sentence encoders (§ 3.1) and sentence extractors (§ 3.2). We compare Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) based sentence representations to the simpler approach of word embedding averaging to understand the gains derived from more sophisticated architectures. We also question the necessity of auto-regressive sentence extraction (i.e. using previous predictions to inform future predictions), which previous approaches have used (§ 2), and propose two alternative models that extract sentences independently.

Our main results (§ 5) reveal:

1. Sentence position bias dominates the learning signal for news summarization, though not for other domains.[2] Summary quality for news is only slightly degraded when content words are omitted from sentence embeddings.
2. Word embedding averaging is as good or better than either RNNs or CNNs for sentence embedding across all domains.
3. Pre-trained word embeddings are as good, or better than, learned embeddings in five of six datasets.
4. Non auto-regressive sentence extraction performs as good or better than auto-regressive extraction in all domains.

Taken together, these and other results in the paper suggest that we are over-estimating the abil-

[1] Data preprocessing and implementation code can be found here: https://github.com/kedz/nnsum/tree/emnlp18-release

[2] This is a known bias in news summarization (Nenkova, 2005).

ity of deep learning models to learn robust and meaningful content features for summarization. In one sense, this might lessen the burden of applying neural network models of content to other domains; one really just needs in-domain word embeddings. However, if we want to learn something other than where the start of the article is, we will need to design other means of sentence representation, and possibly external knowledge representations, better suited to the summarization task.

2 Related Work

The introduction of the CNN-DailyMail corpus by Hermann et al. (2015) allowed for the application of large-scale training of deep learning models for summarization. Cheng and Lapata (2016) developed a sentence extractive model that uses a word level CNN to encode sentences and a sentence level sequence-to-sequence model to predict which sentences to include in the summary. Subsequently, Nallapati et al. (2017) proposed a different model using word-level bidirectional RNNs along with a sentence level bidirectional RNN for predicting which sentences should be extracted. Their sentence extractor creates representations of the whole document and computes separate scores for salience, novelty, and location. These works represent the state-of-the-art for deep learning-based extractive summarization and we analyze them further in this paper.

Other recent neural network approaches include, Yasunaga et al. (2017), who learn a graph-convolutional network (GCN) for multi-document summarization. They do not closely examine the choice of sentence encoder, which is one of the focuses of the present paper; rather, they study the best choice of graph structure for the GCN, which is orthogonal to this work.

Non-neural network learning-based approaches have also been applied to summarization. Typically they involve learning n-gram feature weights in linear models along with other non-lexical word or structural features (Berg-Kirkpatrick et al., 2011; Sipos et al., 2012; Durrett et al., 2016). In this paper, we study representation learning in neural networks that can capture more complex word level feature interactions and whose dense representations are more compatible with current practices in NLP.

The previously mentioned works have focused on news summarization. To further understand the content selection process, we also explore other domains of summarization. In particular, we explore personal narrative summarization based on stories shared on Reddit (Ouyang et al., 2017), workplace meeting summarization (Carletta et al., 2005), and medical journal article summarization (Mishra et al., 2014).

While most work on these summarization tasks often exploit domain-specific features (e.g. speaker identification in meeting summarization (Galley, 2006; Gillick et al., 2009)), we purposefully avoid such features in this work in order to understand the extent to which deep learning models can perform content selection using only surface lexical features. Summarization of academic literature (including medical journals), has long been a research topic in NLP (Kupiec et al., 1995; Elhadad et al., 2005), but most approaches have explored facet-based summarization (Jaidka et al., 2017), which is not the focus of our work.

3 Methods

The goal of extractive text summarization is to select a subset of a document's text to use as a summary, i.e. a short gist or excerpt of the central content. Typically, we impose a budget on the length of the summary in either words or bytes. In this work, we focus on *sentence* extractive summarization, where the basic unit of extraction is a sentence and impose a word limit as the budget.

We model the sentence extraction task as a sequence tagging problem, following (Conroy and O'Leary, 2001). Specifically, given a document containing n sentences $s_1, \ldots, s_n$ we generate a summary by predicting a corresponding label sequence $y_1, \ldots, y_n \in \{0, 1\}^n$, where $y_i = 1$ indicates the i-th sentence is to be included in the summary. Each sentence is itself a sequence of word embeddings $s_i = w_1^{(i)}, \ldots, w_{|s_i|}^{(i)}$ where $|s_i|$ is the length of the sentence in words. The word budget $c \in \mathbb{N}$ enforces a constraint that the total summary word length $\sum_{i=1}^{n} y_i \cdot |s_i| \leq c$.

For a typical deep learning model of extractive summarization there are two main design decisions: *a)* the choice of *sentence encoder* which maps each sentence s_i to an embedding h_i, and *b)* the choice of *sentence extractor* which maps a sequence of sentence embeddings $h = h_1, \ldots, h_n$ to a sequence of extraction decisions $y = y_1, \ldots, y_n$.

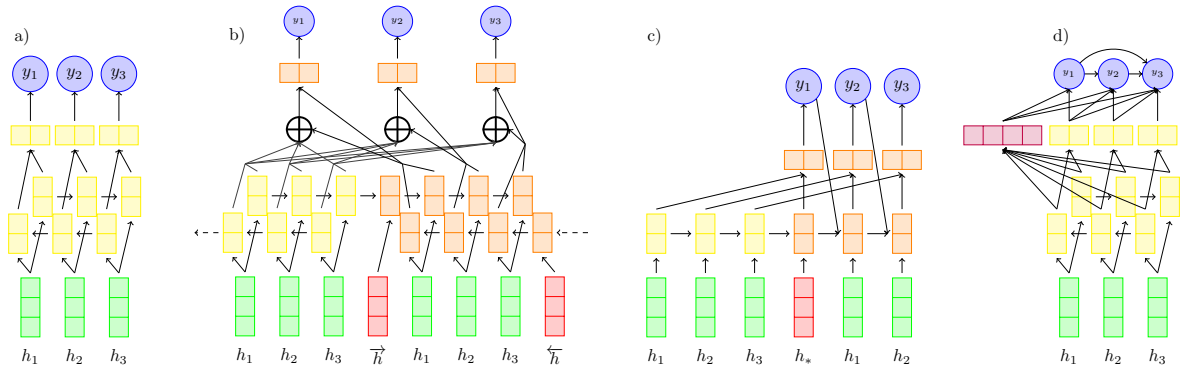

Figure 1: Sentence extractor architectures: a) RNN, b) Seq2Seq, c) Cheng & Lapata, and d) SummaRunner. The $\bigoplus$ indicates attention. Green blocks repesent sentence encoder output and red blocks indicates learned "begin decoding" embeddings. Vertically stacked yellow and orange boxes indicate extractor encoder and decoder hidden states respectively. Horizontal orange and yellow blocks indicate multi-layer perceptrons. The purple blocks represent the document and summary state in the SummaRunner extractor.

3.1 Sentence Encoders

We experiment with three architectures for mapping sequences of word embeddings to a fixed length vector: averaging, RNNs, and CNNs. Hyperparameter settings and implementation details can be found in Appendix A.

Averaging Encoder Under the averaging encoder, a sentence embedding h is simply the average of its word embeddings, i.e. $h = \frac{1}{|s|}\sum_{i=1}^{|s|} w_i$.

RNN Encoder When using the *RNN* sentence encoder, a sentence embedding is the concatenation of the final output states of a forward and backward RNN over the sentence's word embeddings. We use a Gated Recurrent Unit (GRU) for the RNN cell (Chung et al., 2014).

CNN Encoder The *CNN* sentence encoder uses a series of convolutional feature maps to encode each sentence. This encoder is similar to the convolutional architecture of Kim (2014) used for text classification tasks and performs a series of "one-dimensional" convolutions over word embeddings. The final sentence embedding h is a concatenation of all the convolutional filter outputs after max pooling over time.

3.2 Sentence Extractors

Sentence extractors take sentence embeddings $h_{1:n}$ and produce an extract $y_{1:n}$. The sentence extractor is essentially a discriminative classifier $p(y_{1:n}|h_{1:n})$. Previous neural network approaches to sentence extraction have assumed an auto-regressive model, leading to a semi-Markovian factorization of the extractor probabilities $p(y_{1:n}|h) = \prod_{i=1}^{n} p(y_i|y_{<i}, h)$, where each prediction y_i is dependent on *all* previous y_j for all $j < i$. We compare two such models proposed by Cheng and Lapata (2016) and Nallapati et al. (2017). A simpler approach that does not allow interaction among the $y_{1:n}$ is to model $p(y_{1:n}|h) = \prod_{i=1}^{n} p(y_i|h)$, which we explore in two proposed extractor models that we refer to as the RNN and Seq2Seq extractors. Implementation details for all extractors are in Appendix B.

Previously Proposed Sentence Extractors We consider two recent state-of-the-art extractors.

The first, proposed by Cheng and Lapata (2016), is built around a sequence-to-sequence model. First, each sentence embedding[3] is fed into an encoder side RNN, with the final encoder state passed to the first step of the decoder RNN. On the decoder side, the same sentence embeddings are fed as input to the decoder and decoder outputs are used to predict each y_i. The decoder input is weighted by the previous extraction probability, inducing the dependence of y_i on $y_{<i}$. See Figure 1.c for a graphical layout of the extractor.

Nallapati et al. (2017) proposed a sentence extractor, which we refer to as the SummaRunner Extractor, that factorizes the extraction probability into contributions from different sources. First, a bidirectional RNN is run over the sentence em-

[3] Cheng and Lapata (2016) used an CNN sentence encoder with this extractor architecture; in this work we pair the Cheng & Lapata extractor with several different encoders.

beddings[4] and the output is concatenated. A representation of the whole document is made by averaging the RNN output. A summary representation is also constructed by taking the sum of the previous RNN outputs weighted by their extraction probabilities. Extraction predictions are made using the RNN output at the i-th step, the document representation, and i-th version of the summary representation, along with factors for sentence location in the document. The use of the iteratively constructed summary representation creates a dependence of y_i on all $y_{<i}$. See Figure 1.d for a graphical layout.

Proposed Sentence Extractors We propose two sentence extractor models that make a stronger conditional independence assumption $p(y|h) = \prod_{i=1}^{n} p(y_i|h)$, essentially making independent predictions conditioned on h.

RNN Extractor Our first proposed model is a very simple bidirectional RNN based tagging model. As in the RNN sentence encoder we use a GRU cell. The forward and backward outputs of each sentence are passed through a multi-layer perceptron with a logsitic sigmoid output to predict the probability of extracting each sentence. See Figure 1.a for a graphical layout.

Seq2Seq Extractor One shortcoming of the RNN extractor is that long range information from one end of the document may not easily be able to affect extraction probabilities of sentences at the other end. Our second proposed model, the Seq2Seq extractor mitigates this problem with an attention mechanism commonly used for neural machine translation (Bahdanau et al., 2014) and abstractive summarization (See et al., 2017). The sentence embeddings are first encoded by a bidirectional GRU. A separate decoder GRU transforms each sentence into a query vector which attends to the encoder output. The attention weighted encoder output and the decoder GRU output are concatenated and fed into a multi-layer perceptron to compute the extraction probability. See Figure 1.b for a graphical layout.

4 Datasets

We perform our experiments across six corpora from varying domains to understand how different biases within each domain can affect content selection. The corpora come from the news domain (CNN-DailyMail, New York Times, DUC), personal narratives domain (Reddit), workplace meetings (AMI), and medical journal articles (PubMed). See Table 1 for dataset statistics.

Dataset	Train	Valid	Test	Refs
CNN/DM	287,113	13,368	11,490	1
NYT	44,382	5,523	6,495	1.93
DUC	516	91	657	2
Reddit	404	24	48	2
AMI	98	19	20	1
PubMed	21,250	1,250	2,500	1

Table 1: Sizes of the training, validation, test splits for each dataset and the average number of test set human reference summaries per document.

CNN-DailyMail We use the preprocessing and training, validation, and test splits of See et al. (2017). This corpus is a mix of news on different topics including politics, sports, and entertainment.

New York Times The New York Times (NYT) corpus (Sandhaus, 2008) contains two types of abstracts for a subset of its articles. The first summary is an archival abstract and the second is a shorter online teaser meant to entice a viewer of the webpage to click to read more. From this collection, we take all articles that have a concatenated summary length of at least 100 words. We create training, validation, and test splits by partitioning on dates; we use the year 2005 as the validation data, with training and test partitions including documents before and after 2005 respectively.

DUC We use the single document summarization data from the 2001 and 2002 Document Understanding Conferences (DUC) (Over and Liggett, 2002). We split the 2001 data into training and validation splits and reserve the 2002 data for testing.

AMI The AMI corpus (Carletta et al., 2005) is a collection of real and staged office meetings annotated with text transcriptions, along with abstractive summaries. We use the prescribed splits.

[4] Nallapati et al. (2017) use an RNN sentence encoder with this extractor architecture; in this work we pair the SummaRunner extractor with different encoders.

Extractor	Enc.	CNN/DM		NYT		DUC 2002		Reddit		AMI		PubMed	
		M	R-2	M	R-2	M	R-2	M	R-2	M	R-2	M	R-2
Lead	–	24.1	24.4	30.0	32.3	25.1	21.5	**20.1**	**10.9**	12.3	2.0	15.9	9.3
RNN	Avg.	**25.2**	25.4	29.8	34.7	**26.8**	22.7	**20.4**	**11.4**	**17.0**	**5.5**	19.8	17.0
	RNN	25.1	25.4	29.6	34.9	**26.8**	22.6	**20.2**	**11.4**	16.2	**5.2**	19.7	16.6
	CNN	25.0	25.1	29.0	33.7	**26.7**	**22.7**	**20.9**	**12.8**	14.4	3.2	19.9	16.8
Seq2Seq	Avg.	**25.2**	**25.6**	**30.5**	**35.7**	**27.0**	**22.8**	**20.9**	**13.6**	**17.0**	**5.5**	**20.1**	**17.7**
	RNN	**25.1**	25.3	30.2	**35.9**	**26.7**	22.5	**20.5**	**12.0**	16.1	**5.3**	19.7	16.7
	CNN	25.0	25.1	29.9	35.1	**26.7**	**22.7**	**20.7**	**13.2**	14.2	2.9	19.8	16.9
Cheng & Lapata	Avg.	25.0	25.3	30.4	**35.6**	**27.1**	**23.1**	**20.9**	**13.6**	**16.7**	**6.1**	**20.1**	**17.7**
	RNN	25.0	25.0	**30.3**	**35.8**	**27.0**	**23.0**	**20.3**	**12.6**	**16.3**	**5.0**	19.7	16.7
	CNN	**25.2**	25.1	29.9	35.0	**26.9**	**23.0**	**20.5**	**13.4**	14.3	2.8	19.9	16.9
Summa Runner	Avg.	25.1	25.4	30.2	35.4	26.7	22.3	**21.0**	**13.4**	**17.0**	**5.6**	19.9	17.2
	RNN	25.1	25.2	30.0	35.5	26.5	22.1	**20.9**	**12.5**	**16.5**	**5.4**	19.7	16.5
	CNN	24.9	25.0	29.3	34.4	26.4	22.2	**20.4**	**12.3**	14.5	3.2	19.8	16.8
Oracle	–	31.1	36.2	35.3	48.9	31.3	31.8	24.3	16.2	8.1	3.9	24.1	25.0

Table 2: METEOR (M) and ROUGE-2 recall (R-2) results across all extractor/encoder pairs. Results that are statistically indistinguishable from the best system are shown in bold face.

Reddit Ouyang et al. (2017) collected a corpus of personal stories shared on Reddit[5] along with multiple extractive and abstractive summaries. We randomly split this data using roughly three and five percent of the data validation and test respectively.

PubMed We created a corpus of 25,000 randomly sampled medical journal articles from the PubMed Open Access Subset[6]. We only included articles if they were at least 1000 words long and had an abstract of at least 50 words in length. We used the article abstracts as the ground truth human summaries.

4.1 Ground Truth Extract Summaries

Since we do not typically have ground truth extract summaries from which to create the labels y_i, we construct gold label sequences by greedily optimizing ROUGE-1, using the algorithm in Appendix C. We choose to optimize for ROUGE-1 rather than ROUGE-2 similarly to other optimization based approaches to summarization (Sipos et al., 2012; Durrett et al., 2016) which found this to be the easier target to learn.

[5] www.reddit.com

[6] https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/

5 Experiments

We evaluate summary quality using ROUGE-2 recall (Lin, 2004); ROUGE-1 and ROUGE-LCS trend similarity in our experiments. We use target word lengths of 100 words for news, and 75, 290, and 200 for Reddit, AMI, and PubMed respectively. We also evaluate using METEOR (Denkowski and Lavie, 2014).[7] Summaries are generated by extracting the top ranked sentences by model probability $p(y_i = 1|y_{<i}, h)$, stopping when the word budget is met or exceeded. We estimate statistical significance by averaging each document level score over the five random initializations. We then test the difference between the best system on each dataset and all other systems using the approximate randomization test (Riezler and Maxwell, 2005) with the Bonferroni correction for multiple comparisons, testing for significance at the 0.05 level.

5.1 Training

We train all models to minimize the weighted negative log-likelihood

$$\mathcal{L} = - \sum_{\substack{s,y \in \mathcal{D} \\ h = \text{enc}(s)}} \sum_{i=1}^{n} \omega(y_i) \log p\left(y_i | y_{<i}, h\right)$$

[7] We use the default settings for METEOR and use remove stopwords and no stemming options for ROUGE, keeping defaults for all other parameters.

Ext.	Emb.	CNN/DM		NYT		DUC		Reddit		AMI		PubMed	
Seq2Seq	Fixed	**25.6**		**35.7**		**22.8**		**13.6**		5.5		**17.7**	
	Learn	25.3	(0.3)	**35.7**	(0.0)	**22.9**	(-0.1)	**13.8**	(-0.2)	**5.8**	(-0.3)	16.9	(0.8)
C&L	Fixed	**25.3**		**35.6**		**23.1**		**13.6**		**6.1**		**17.7**	
	Learn	24.9	(0.4)	35.4	(0.2)	**23.0**	(0.1)	**13.4**	(0.2)	**6.2**	(-0.1)	16.4	(1.3)
Summa Runner	Fixed	**25.4**		**35.4**		**22.3**		**13.4**		**5.6**		**17.2**	
	Learn	25.1	(0.3)	35.2	(0.2)	**22.2**	(0.1)	12.6	(0.8)	**5.8**	(-0.2)	16.8	(0.4)

Table 3: ROUGE-2 recall across sentence extractors when using fixed pretrained embeddings or when embeddings are updated during training. In both cases embeddings are initialized with pretrained GloVe embeddings. All extractors use the averaging sentence encoder. When both learned and fixed settings are bolded, there is no signifcant performance difference. RNN extractor is omitted for space but is similar to Seq2Seq. Difference in scores shown in parenthesis.

Ablation	CNN/DM	NYT	DUC	Reddit	AMI	PubMed
all words	**25.4**	**34.7**	22.7	**11.4**	5.5	**17.0**
-nouns	25.3† (0.1)	34.3† (0.4)	22.3† (0.4)	10.3† (1.1)	3.8† (1.7)	15.7† (1.3)
-verbs	25.3† (0.1)	34.4† (0.3)	22.4† (0.3)	10.8 (0.6)	5.8 (-0.3)	16.6† (0.4)
-adj/adv	25.3† (0.1)	34.4† (0.3)	22.5 (0.2)	9.5† (1.9)	5.4 (0.1)	16.8† (0.2)
-function	25.2† (0.2)	34.5† (0.2)	**22.9**† (-0.2)	10.3† (1.1)	**6.3**† (-0.8)	16.6† (0.4)

Table 4: ROUGE-2 recall after removing nouns, verbs, adjectives/adverbs, and function words. Ablations are performed using the averaging sentence encoder and the RNN extractor. Bold indicates best performing system. † indicates significant difference with the non-ablated system. Difference in score from *all words* shown in parenthesis.

over the training data $\mathcal{D}$ using stochastic gradient descent with the ADAM optimizer (Kingma and Ba, 2014). $\omega(0) = 1$ and $\omega(1) = N_0/N_1$ where N_y is the number of training examples with label y. We trained for a maximum of 50 epochs and the best model was selected with early stopping on the validation set according to ROUGE-2. Each epoch constitutes a full pass through the dataset. The average stopping epoch was: CNN-DailyMail, 16.2; NYT, 21.36; DUC, 37.11; Reddit, 36.59; AMI, 19.58; PubMed, 19.84. All experiments were repeated with five random initializations. Unless specified, word embeddings were initialized using pretrained GloVe embeddings (Pennington et al., 2014) and we did not update them during training. Unknown words were mapped to a zero embedding. See Appendix D for more optimization and training details.

5.2 Baselines

Lead As a baseline we include the lead summary, i.e. taking the first x words of the document as summary, where x is the target summary length for each dataset (see the first paragraph of § 5). While incredibly simple, this method is still a competitive baseline for single document summarization, especially on newswire.

Oracle To measure the performance ceiling, we show the ROUGE/METEOR scores using the extractive summary which results from greedily optimizing ROUGE-1. I.e., if we had clairvoyant knowledge of the human reference summary, the oracle system achieves the (approximate) maximum possible ROUGE scores. See Appendix C for a detailed description of the oracle algorithm.

5.3 Results

The results of our main experiment comparing the different extractors/encoders are shown in Table 2. Overall, we find no major advantage when using the CNN and RNN sentence encoders over the averaging encoder. The best performing encoder/extractor pair either uses the averaging encoder (five out of six datasets) or the differences are not statistically significant.

When looking at extractors, the Seq2Seq extractor is either part of the best performing system (three out of six datasets) or is not statistically distinguishable from the best extractor.

Overall, on the news and medical journal domains, the differences are quite small with the dif-

Ext.	Order	CNN/DM		NYT		DUC		Reddit		AMI		PubMed	
Seq2Seq	In-Order	**25.6**		**35.7**		**22.8**		**13.6**		5.5		**17.7**	
	Shuffled	21.7	(3.9)	25.6	(10.1)	21.2	(1.6)	**13.5**	(0.1)	**6.0**	(-0.5)	14.9	(2.8)

Table 5: ROUGE-2 recall using models trained on in-order and shuffled documents. Extractor uses the averaging sentence encoder. When both in-order and shuffled settings are bolded, there is no signifcant performance difference. Difference in scores shown in parenthesis.

Hurricane Gilbert swept toward the Dominican Republic Sunday, and the Civil Defense alerted its heavily populated south coast to prepare for high winds, heavy rains and high seas. The storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph. An estimated 100,000 people live in the province, including 70,000 in the city of Barahona, about 125 miles west of Santo Domingo. **On Saturday, Hurricane Florence was downgraded to a tropical storm and its remnants pushed inland from the U.S. Gulf Coast.** Tropical Storm Gilbert formed in the eastern Caribbean and strengthened into a hurricane Saturday night.	Hurricane Gilbert swept toward the Dominican Republic Sunday, and the Civil Defense alerted its heavily populated south coast to prepare for high winds, heavy rains and high seas. The storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph. An estimated 100,000 people live in the province, including 70,000 in the city of Barahona, about 125 miles west of Santo Domingo. Tropical Storm Gilbert formed in the eastern Caribbean and strengthened into a hurricane Saturday night. **Strong winds associated with the Gilbert brought coastal flooding, strong southeast winds and up to 12 feet feet to Puerto Rico's south coast.**

Table 6: Example output of Seq2Seq extractor (left) and Cheng & Lapata Extractor (right). This is a typical example, where only one sentence is different between the two (shown in bold).

ferences between worst and best systems on the CNN/DM dataset spanning only .56 of a ROUGE point. While there is more performance variability in the Reddit and AMI data, there is less distinction among systems: no differences are significant on Reddit and every extractor has at least one configuration that is indistinguishable from the best system on the AMI corpus. This is probably due to the small test size of these datasets.

Word Embedding Learning Given that learning a sentence encoder (averaging has no learned parameters) does not yield significant improvement, it is natural to consider whether learning word embeddings is also necessary. In Table 3 we compare the performance of different extractors using the averaging encoder, when the word embeddings are held fixed or learned during training. In both cases, word embeddings are initialized with GloVe embeddings trained on a combination of Gigaword and Wikipedia. When learning embeddings, words occurring fewer than three times in the training data are mapped to an unknown token (with learned embedding).

In all but one case, fixed embeddings are as good or better than the learned embeddings. This is a somewhat surprising finding on the CNN/DM data since it is reasonably large, and learning embeddings should give the models more flexibility to identify important word features.[8] This suggests that we cannot extract much generalizable learning signal from the content other than what is already present from initialization. Even on PubMed, where the language is quite different from the news/Wikipedia articles the GloVe embeddings were trained on, learning leads to significantly worse results.

POS Tag Ablation It is also not well explored what word features are being used by the encoders. To understand which classes of words were most important we ran an ablation study, selectively removing nouns, verbs (including participles and auxiliaries), adjectives & adverbs, and function words (adpositions, determiners, conjunctions). All datasets were automatically tagged using the spaCy part-of-speech (POS) tagger[9]. The embeddings of removed words were replaced with a zero vector, preserving the order and position of the non-ablated words in the sentence. Ablations were performed on training, validation, and test partitions, using the RNN extractor with averaging encoder. Table 4 shows the results of the POS tag ablation experiments. While removing any word class from the representation generally hurts performance (with statistical significance), on the news domains, the absolute values of the

[8]The AMI corpus is an exception here where learning *does* lead to small performance boosts, however, only in the Seq2Seq extractor is this diference significant; it is quite possible that this is an artifact of the very small test set size.

[9]https://github.com/explosion/spaCy

differences are quite small (.18 on CNN/DM, .41 on NYT, .3 on DUC) suggesting that the model's predictions are not overly dependent on any particular word types. On the non-news datasets, the ablations have a larger effect (max differences are 1.89 on Reddit, 2.56 on AMI, and 1.3 on PubMed). Removing nouns leads to the largest drop on AMI and PubMed. Removing adjectives and adverbs leads to the largest drop on Reddit, suggesting the intensifiers and descriptive words are useful for identifying important content in personal narratives. Curiously, removing the function word POS class yields a significant improvement on DUC 2002 and AMI.

Document Shuffling Sentence position is a well known and powerful feature for news summarization (Hong and Nenkova, 2014), owing to the intentional lead bias in the news article writing[10]; it also explains the difficulty in beating the lead baseline for single-document summarization (Nenkova, 2005; Brandow et al., 1999). In examining the generated summaries, we found most of the selected sentences in the news domain came from the lead paragraph of the document. This is despite the fact that there is a long tail of sentence extractions from later in the document in the ground truth extract summaries (31%, 28.3%, and 11.4% of DUC, CNN/DM, and NYT training extract labels come from the second half of the document). Because this lead bias is so strong, it is questionable whether the models are learning to identify important content or just find the start of the document. We conduct a sentence order experiment where each document's sentences are randomly shuffled during training. We then evaluate each model performance on the unshuffled test data, comparing to the model trained on unshuffled data; if the models trained on shuffled data drop in performance, then this indicates the lead bias is the relevant factor.

Table 5 shows the results of the shuffling experiments. The news domains and PubMed suffer a significant drop in performance when the document order is shuffled. By comparison, there is no significant difference between the shuffled and in-order models on the Reddit domain, and shuffling actually improves performance on AMI. This suggest that position is being learned by the models in the news/journal article domain even when the model has no explicit position features, and that this feature is more important than either content or function words.

[10] https://en.wikipedia.org/wiki/Inverted_pyramid_(journalism)

6 Discussion

Learning content selection for summarization in the news domain is severely inhibited by the lead bias. The summaries generated by all systems described here–the prior work and our proposed simplified models–are highly similar to each other and to the lead baseline. The Cheng & Lapata and Seq2Seq extractors (using the averaging encoder) share 87.8% of output sentences on average on the CNN/DM data, with similar numbers for the other news domains (see Table 6 for a typical example). Also on CNN/DM, 58% of the Seq2Seq selected sentences also occur in the lead summary, with similar numbers for DUC, NYT, and Reddit. Shuffling reduces lead overlap to 35.2% but the overall system performance drops significantly; the models are not able to identify important information without position.

The relative robustness of the news domain to part of speech ablation also suggests that models are mostly learning to recognize the stylistic features unique to the beginning of the article, and not the content. Additionally, the drop in performance when learning word embeddings on the news domain suggests that word embeddings alone do not provide very generalizable content features compared to recognizing the lead.

The picture is rosier for non-news summarization where part of speech ablation leads to larger performance differences and shuffling either does not inhibit content selection significantly or leads to modest gains. Learning better word-level representations on these domains will likely require much larger corpora, something which might remain unlikely for personal stories and meetings.

The lack of distinction among sentence encoders is interesting because it echoes findings in the generic sentence embedding literature where word embedding averaging is frustratingly difficult to outperform (Iyyer et al., 2015; Wieting et al., 2015; Arora et al., 2016; Wieting and Gimpel, 2017). The inability to learn useful sentence representations is also borne out in the SummaRunner model, where there are explicit similarity computations between document or summary representations and sentence embeddings; these computations do not seem to add much to the per-

formance as the Cheng & Lapata and Seq2Seq models which lack these features generally perform as well or better. Furthermore, the Cheng & Lapata and SummaRunner extractors both construct a history of previous selection decisions to inform future choices but this does not seem to significantly improve performance over the Seq2Seq extractor (which does not). This suggests that we need to rethink or find novel forms of sentence representation for the summarization task.

A manual examination of the outputs revealed some interesting failure modes, although in general it was hard to discern clear patterns of behaviour other than lead bias. On the news domain, the models consistently learned to ignore quoted material in the lead, as often the quotes provide color to the story but are unlikely to be included in the summary (e.g. *"It was like somebody slugging a punching bag."*). This behavior was most likely triggered by the presence of quotes, as the quote attributions, which were often tokenized as separate sentences, would subsequently be included in the summary despite also not containing much information (e.g. *Gil Clark of the National Hurricane Center said Thursday*).

7 Conclusion

We have presented an empirical study of deep learning based content selection algorithms for summarization. Our findings suggest such models face stark limitations on their ability to learn robust features for this task and that more work is needed on sentence representation for summarization.

8 Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable feedback. Thanks goes out as well to Chris Hidey for his helpful comments.

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via contract # FA8650-17-C-9117. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2016. A simple but tough-to-beat baseline for sentence embeddings.

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

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 481–490. Association for Computational Linguistics.

Ronald Brandow, Karl Mitze, and Lisa Rau. 1999. Automatic condensation of electronic publications by sentence selection. In Jan Fagerberg, David C. Mowery, and Richard R. Nelson, editors, *Advances in Automatic Text Summarization*, chapter 19, pages 293–303. MIT Press, Oxford.

Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, et al. 2005. The ami meeting corpus: A pre-announcement. In *International Workshop on Machine Learning for Multimodal Interaction*, pages 28–39. Springer.

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

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

John M Conroy and Dianne P O'Leary. 2001. Text summarization via hidden markov models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 406–407. ACM.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. *arXiv preprint arXiv:1603.08887*.

Noemie Elhadad, M-Y Kan, Judith L Klavans, and KR McKeown. 2005. Customization in a unified framework for summarizing medical literature. *Artificial intelligence in medicine*, 33(2):179–198.

Michel Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 364–372. Association for Computational Linguistics.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Dan Gillick, Korbinian Riedhammer, Benoit Favre, and Dilek Hakkani-Tur. 2009. A global optimization framework for meeting summarization. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 4769–4772. IEEE.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256.

Karl Moritz Hermann, Tomáš Kočiský, 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 (NIPS)*.

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

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1681–1691.

Kokil Jaidka, Muthu Kumar Chandrasekaran, Sajal Rustagi, and Min-Yen Kan. 2017. Insights from cl-scisumm 2016: the faceted scientific document summarization shared task. *International Journal on Digital Libraries*, pages 1–9.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

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

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73. ACM.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Rashmi Mishra, Jiantao Bian, Marcelo Fiszman, Charlene R Weir, Siddhartha Jonnalagadda, Javed Mostafa, and Guilherme Del Fiol. 2014. Text summarization in the biomedical domain: a systematic review of recent research. *Journal of biomedical informatics*, 52:457–467.

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. 2005. Automatic text summarization of newswire: Lessons learned from the document understanding conference. In *AAAI*, volume 5, pages 1436–1441.

Jessica Ouyang, Serina Chang, and Kathy McKeown. 2017. Crowd-sourced iterative annotation for narrative summarization corpora. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 46–51.

Paul Over and Walter Liggett. 2002. Introduction to duc: An intrinsic evaluation of generic news text summarization systems. *Proc. DUC. http://wwwnlpir. nist. gov/projects/duc/guidelines/2002. html*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Stefan Riezler and John T Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing for mt. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 57–64.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Ruben Sipos, Pannaga Shivaswamy, and Thorsten Joachims. 2012. Large-margin learning of submodular summarization models. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 224–233. Association for Computational Linguistics.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*.

John Wieting and Kevin Gimpel. 2017. Revisiting recurrent networks for paraphrastic sentence embeddings. *arXiv preprint arXiv:1705.00364*.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based neural multi-document summarization. *arXiv preprint arXiv:1706.06681*.