Abstractive and Extractive Text Summarization using Document Context Vector and Recurrent Neural Networks

Chandra Khatri*
Amazon Lab126
Sunnyvale, California
ckhatri@amazon.com

Gyanit Singh
eBay Inc.
San Jose, California
gysingh@ebay.com

Nish Parikh †
Google
Mountain View, California
nishparikh@google.com

ABSTRACT

Sequence to sequence (Seq2Seq) learning has recently been used for abstractive and extractive summarization. In current study, Seq2Seq models have been used for eBay product description summarization. We propose a novel Document-Context based Seq2Seq models using RNNs for abstractive and extractive summarizations. Intuitively, this is similar to humans reading the title, abstract or any other contextual information before reading the document. This gives humans a high-level idea of what the document is about. We use this idea and propose that Seq2Seq models should be started with contextual information at the first time-step of the input to obtain better summaries. In this manner, the output summaries are more document centric, than being generic, overcoming one of the major hurdles of using generative models. We generate document-context from user-behavior and seller provided information. We train and evaluate our models on human-extracted-golden-summaries. The document-contextual Seq2Seq models outperform standard Seq2Seq models. Moreover, generating human extracted summaries is prohibitively expensive to scale, we therefore propose a semi-supervised technique for extracting approximate summaries and using it for training Seq2Seq models at scale. Semi-supervised models are evaluated against human extracted summaries and are found to be of similar efficacy. We provide side by side comparison for abstractive and extractive summarizers (contextual and non-contextual) on same evaluation dataset. Overall, we provide methodologies to use and evaluate the proposed techniques for large document summarization. Furthermore, we found these techniques to be highly effective, which is not the case with existing techniques.

CCS CONCEPTS

• **Information systems** → **Summarization**; • **Computing methodologies** → **Neural networks**; *Semi-supervised learning settings*;

KEYWORDS

Text Summarization; Recurrent Neural Networks; Natural Language Processing; Information Retrieval; Extraction; Abstraction; Language Modeling; Topic Signature; Deep Learning; e-Commerce

---

*Work was done while the author was at eBay Inc.
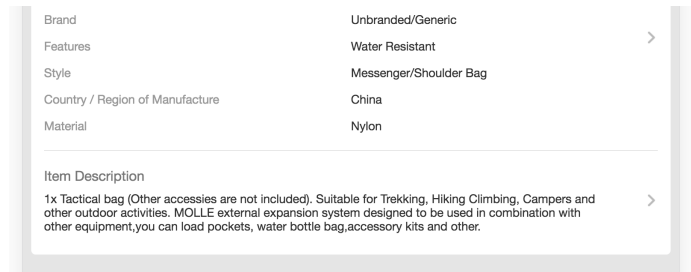†Work was done while the author was at eBay Inc.

Figure 1: Snapshot of a product snippet as it appears on the eBay mobile app for product titled *"Mens Travel Hiking Military Tactical Army Camo Sling Backpack Chest Shoulder Bag". The full html description is hidden behind the click (arrow on "Item description") making the current page easy to consume and user friendly.*

1 INTRODUCTION

Document summarization has its applications in almost all the domains of the Internet. Search engines provide query and context specific summary snippets as a part of search experience, news websites use summaries to brief the articles, social media use them for content targeting while e-commerce websites use summaries for better browse experience through item or product highlights. In this paper, we leverage data-sets from a popular B2C and C2C eBay. Presenting users with product summary decreases user's cognitive load to evaluate product's relevance to their purchase intent, leading to higher engagement and better browsing experience. Furthermore, as the traffic on mobile sites and applications is increasing, item summaries become much more relevant. Due to limited real-estate available for mobile sites and design point of view it is more relevant to show item summaries than the entire HTML elements. Figure 1 depicts the picture of the snippet on eBay mobile application.

Text summarization techniques are either extractive or abstractive. In extraction, key sentences and objects are extracted without modifying the objects themselves. This is obtained by key-phrase or ad-hoc sentence extraction keeping the sentences intact [2, 3]. While abstraction involves paraphrasing the context-aware sentences after understanding the language [4, 32]. Abstractive techniques generally requires large-scale data (documents and corresponding summaries) for training the models, for example news titles can be considered as summaries and the articles can be considered as the document. In many instances, while summarizing content generated by third party, web systems have a legal constraints on modification of the content. In these cases, summaries

are extracted than being generated. Since, majority of the content written for products in marketplace is provided by the sellers, the marketplace also has legal constraint about not revising the content.

In this work, we propose a technique for generation and incorporating *document context* in Seq2Seq based generative models for abstractive and extractive summarization. Using document-context is akin to humans reading title and abstract to know the key-details before delving into the full document. In this paper we describe techniques for generating context around documents using user behavior and other information provided by the document creators. We show that RNNs for both abstraction and extraction, both benefit from feeding document context at the first time step of sequence to sequence learning. We evaluate the summaries generated by this methodology and found them to be more contextual to the document and preferred by humans.

Human extracted summaries are used for training extractive models - like extractive-RNNs. However, this approach for training is not scalable since extracting summaries for millions/hundreds of thousands of documents is prohibitively expensive. Sequence to sequence model generally perform best with large scale data. Therefore we proposed a novel approach using document-context for extracting approximate summaries in a semi-supervised fashion, which is used for large scale training.

Abstractive sequence to sequence models are generally trained on titles and subtitles. We adopt a similar approach with document-context, which helps us scaling the training. Furthermore, we perform a heuristics by using the trained model for re-ranking all the sentences (within the document) based on likelihood of a sentence in a document being a summary sentence during the inference. We adopted the idea from a state-of-the-art techniques for Automatic Speech Recognition, wherein an RNN is used for re-ranking the potential outcomes from an n-gram based language models [40].

Following are the main contributions of our work:-

- Obtaining the context vectors from documents, which can be used for extractive and abstractive summarization tasks.
- Automatically extracting approximate summaries and creating training data to enable large scale semi-supervised learning for extractive summarization. This is shown to be competitive to supervised learning techniques.
- Using RNN and CNN-RNN for extractive summarization.
- Abstractive Seq2Seq summarization for large documents.
- Using document-context vectors for improving the Seq2Seq learning for text summarization. This novel approach is shown to beat the state of the art in a similar settings.
- Comparing abstractive to extractive summarization under same setting of sequence to sequence learning.

2 PREVIOUS WORK

Several summarization techniques have been explored over the past decades [26, 35] following are some of the popular techniques:

Surface level approaches consider title words and cue-words (e.g. "important", "best" etc.) for extracting relevant sentences [22, 29].

Corpus based approaches leverage structural distribution of words using internal or external corpus (e.g. WordNet [25]) for summarization.

Cohesion based approaches considers cohesive relations between the concepts within the text such as (antonyms, repetitions, synonyms etc.) using Lexical Chains [3]

Graph based approaches are some of the most popular text summarization techniques. Each sentence in the text is represented as a vertex and a graph is constructed around all the sentences, where the edges correspond to the inter-connections between the sentences. LexRank [11] and TextRank [23] are two such techniques.

Machine learning based approaches: Document summarization can be converted to a supervised or semi-supervised learning problem. In supervised learning approaches, hints or clues such as key-phrases, topic words, blacklist words, are used to label the sentences as positive or negative classes or the sentences are manually tagged (which is not scalable). Once the labels are established, a binary classier can be trained for obtaining the scores or summary-likelihood score pertaining to each sentence. Several summarization techniques have been explored in the literature [9] in this regard. However, classification based approaches generalize well, however they are not efficient in extracting document specific summaries. Training data for machine learning approach contains label of the sentences irrespective of the document. If the document level information is not provided then these approaches provide same prediction irrespective of the document. Providing document context in the models alleviates this problem, which is what is one of the contributions of this paper.

Abstractive summarization techniques are less prevalent in the literature than the extractive ones. It is much harder because it involves re-writing the sentences which if performed manually, is not scalable and requires natural language generation techniques. The two common abstraction techniques are structured and semantic [26, 32], both of which mostly are either graph/tree based or ontology and rule (e.g. template) based.

All the approaches mentioned above work well; however, they either face challenge towards scalability or large scale evaluation or do not generalize well. Due to complexity constraints, research to date has focused primarily on extractive methods, but due to advancements in Seq2Seq and Natural Language Generation techniques is making it possible to generate reasonable summaries for very short descriptions using Abstraction [1, 4, 7, 15, 19, 21, 27, 28, 32, 41]. [30] used reinforcement learning while [12] used CNNs for abstractive summarization. Most of the current advancements are around short summaries from short documents. Furthermore, there does not exist any work where extractive and abstractive techniques have been compared in the similar setting using Seq2Seq approaches.

Seq2Seq techniques based approaches have been used to efficiently map the input sequences (description / document) to map output sequence (summary), however they require large amounts data. With several examples, model tends to learn the mapping between input sequence and output sequence and generate more efficient summaries corresponding to the input document. Moreover, it is found that Seq2Seq models currently work well for smaller document summarizations (one-two lines of the document mapping to headlines/phrase representation) [1, 7]. Even though Seq2Seq models are providing benchmark results in Machine Translation and Speech Recognition tasks [6, 36, 39] they have not yet performed

well for summarization tasks, dialog systems and evaluation of dialog systems [14, 31, 38] and are facing many challenges (e.g. summarizing long documents). For long sequence [36] depicted that reversing the source sentence provides better results.

Architecturally our abstractive models are similar to [7], with a change in encoder being novel Document-Contextual-LSTM instead of simple attention-based mechanism. Document-Contextual-Vector is described in Section 3. Furthermore, unlike related previous work, models in current study are summarizing full/large documents (longer than 2500 characters) to generate relatively large summaries (800 characters) using Beam-search and vocabulary restrictions constraints. Furthermore, we have not found any literature where Seq2Seq based abstractive summarization is compared with related RNN/CNN variations for extractive summarization, which is novel addition of this paper. Khatri et. al. [16–18] describes summarization for eCommerce setting.

3 USER ACTIONS TO INFER DOCUMENT CONTEXT

Websites which host user generated documents for consumption by other users provide myriads of ways for document discovery. For example, users on eCommerce website, discover relevant document to their intent via search, recommendation modules or various topical pages. Also, sellers or content creators while creating the document or product pages provide lot of metadata on the document, for example title, tags, categorical, etc. In this section we describe how we use historical information from document creator and document consumer to generate the document context. This context is then later combined with word embeddings obtained via Skip-Gram Model with Negative Sampling (SGNS) [24] to generate what we call *document context vectors* or DCV. We also use document context to score sentences in the document for algorithmic labeling for large scale semi-supervised learning to obtain approximate summaries. Stop words and high frequency words in vocabulary are not considered while obtaining the context words in the following subsections.

Let $\mathbb{V}$ be the vocabulary of the corpora. We assume that stop words have been removed from the corpora to generate the vocabulary. Also let N be the vocabulary size i.e. $|\mathbb{V}| = N$. For the document d, we generate three *unit-vectors* namely $\widehat{C_s^d}$ (s denotes seller), $\widehat{C_q^d}$ (q denotes queries), $\widehat{C_b^d}$ (b denotes browse). All of these vectors have dimension N, but for document d most of the dimensions are 0. We later combine these vectors with SGNS vectors to make DCV. For brevity we drop the d from superscript is rest of the section as vectors are being generated per document only.

3.1 Document creators

When a seller lists a product for sale on eBay they create a web-document for it, which includes title, key-value meta-data on fixed dimensions like condition, brand, size, color etc. They also type in verbose description containing images, videos and html. In addition to this they select a leaf in the eBay taxonomy, $C = \{l_1, l_2, \ldots, l_n\}$ where l_i is taxon used in the taxonomy. The taxonomy used in eBay is a laminar family. That is given two taxa l_1 and l_2 either $l_1 \cap l_2 = \phi$
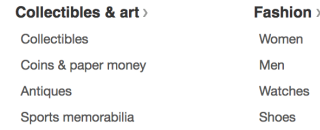
Figure 2: A snapshot of taxonomy used in a eBay. Here the top taxa Collectibles and Art are expanded into lower taxa for example Antiques.

or $l_1 \subset l_2$ or $l_2 \subset l_1$. Seller has to attach a leaf in this taxonomy tree to the document.

Figure 2 shows a small snapshot of eBay taxonomy. This taxonomy tree is maintained and generated by domain experts and is of high quality. For example the document titled *"Mens Travel Hiking Military Tactical Army Camo Sling Backpack Chest Shoulder Bag"* is chosen to be put in the following leaf *Sporting Goods > Outdoor Sports > Camping & Hiking > Hiking Backpacks > Day Packs* by the seller.

Context vector C_s is induced from seller provided metadata m_s = {*title, subtitle, taxon, other key-valued metadata such as brand, color etc.*}. For $w \in \mathbb{V}$ let $C_s(w)$ be the value of the dimension w in C_s. It is defined as follows,

$$C_s(w) = \begin{cases} frequency(w) \text{ in } m_s, & \text{if } w \in \text{ seller metadata} \\ 0, & \text{otherwise} \end{cases}$$

We then normlaize the vector C_s to a unit vector $\widehat{C_s}$. We use the weight obtained for each context word later on to obtain the document context vector for each document giving relevance to words based on weights. This technique is not limited to eBay or any eCommerce based pages only. Such an approach can be easily extended to the general textual documents available online in the form of webpages (e.g.: title, metadata, etc.) or articles (e.g.: title, sub-titles, abstract, keywords, category, etc.)

3.2 Document consumers

Buyers discover relevant inventory via search, recommendations made to them or on topical pages (e.g. http://www.ebay.com/sch/Hiking-Backpacks-Bags/181378/bn_7407781/i.html & https://www.amazon.com/hiking-backpacks-and-bags/b?ie=UTF8&node=3400391) or from external sources like advertising etc. These discovery browse paths whether they are search trails or click trails [33] that ends in a document are used to extract relevant words for the document. One such discovery path is search. For example, when the user searches with the queries like *"tactical sling backpack"*, *"military backpack"* and lands on the document titled *"Mens Travel Hiking Military Tactical Army Camo Sling Backpack Chest Shoulder Bag"* the words of the queries contains what the user thought was the most descriptive information from the document. When such information is aggregated statistically cross large number of users it can provide great context for the document.

Let, $qset^d$ = { queries used to discover document d} and $browse^d$ = { titles of documents via which user discovered document d (via recommendation or topical page) }. For the dimension associated

with word w values in both vectors are defined as

$$C_q(w) = \begin{cases} frequency(w) \text{ in } qset^d, & \text{if } w \in qset^d \\ 0, & \text{otherwise} \end{cases}$$

$$C_b(w) = \begin{cases} frequency(w) \text{ in } browse^d, & \text{if } w \in browse^d \\ 0, & \text{otherwise} \end{cases}$$

We then normalize the vector C_b, C_q to a unit vector $\widehat{C_q}, \widehat{C_b}$. Similar to the weights obtained for the context words from document creators, we obtain the weights for words provided by the document consumers/readers. Basically, there are significant information provided by the readers based on their experience and behavior. We can leverage the information provided by the readers to further obtain more context words and corresponding weights, which is later on used to obtain the document context vector.

3.3 Document Context Vectors

For a document d we defined three unit vectors above $\widehat{C_s^d}$ (s denotes seller), $\widehat{C_q^d}$ (q denotes queries), $\widehat{C_b^d}$ (b denotes browse). We combine these three to form a cumulative context unit vector C^d by adding these three vector.

$$C^d = \beta_s * \widehat{C_s^d} + \beta_q * \widehat{C_q^d} + \beta_b * \widehat{C_b^d}$$

We used $\beta_s = 1, \beta_q = 1$ and $\beta_b = 1$. These parameters may be fine tuned based on historical demand, which may be estimated by ratio of traffic volumes in different channels or by expert set pre-existing priors for importance of various channels of traffic. We then re-weight value for each word dimension (w) in C^d by its idf to make C_{idf}^d. More precisely value for dimension for word w is defined as

$$C_{idf}^d(w) = C^d(w) * idf(w) \text{ for word } w$$

We combine context with word embeddings obtained via SGNS to generate a vector per document that we call v_d. Let M_{SGNS} be the matrix of dimension Nxk where k is the dimension of word embeddings and N is the vocabulary size of the corpora. Row for word w in M_{SGNS} is the word embedding of the word w. We define v_d as

$$v_d = C_{idf}^d * M_{SGNS} \tag{1}$$

Note that dimension of v_d is $1xk$ just like word embeddings.

3.4 Scoring Sentences Using Document Context

For a document d let's say it contains $Sentneces_d = \{$ set of sentences in $d\}$. We describe how to use document context to score these sentences. These score are used for generating algorithmic labels for large-scale semi-supervised learning to obtain approximate summaries. For more details see Section 5.1. For a sentence $S \in Sentneces_d$, where $S = w_1 w_2 \dots w_k$.

$$S_{score} = \sum_{w \in S} v_d(w)$$

The above score corresponds to the weighted sum for the words in the sentence. The weight for the words incorporate frequencies

in seller provided metadata, buyer's search history, browse history (leading to discovery of the said product) and also the Inverse Document Frequence (IDF) score of that word. The IDF score for each word is simply the inverse of documents containing this word across all the documents at eBay. The IDF score corresponds to the topical or document level relevance for each word in the vocabulary. Highly common words such as stop words ("the", "and", etc. have nearly zero IDF score). Using the score above, we obtain the score of each sentence. We use the sentence score to rank each sentence within the document and select top k (till we reach 800 characters) sentences as the summary for a document/product. We did an A/B test to see if the summaries obtained using this technique correlates with user expectations and we observed statistically significant lift in sales and user-engagement. This implies that the summaries obtained using this semi-supervised approach are useful. We use this technique to generate the training data for supervised extraction based summarization techniques (e.g. RNN).

4 MODELS

This work primarily focuses on adding context as the initial state to RNNs for abstractive and extractive text summarizations and comparing it with various state-of-the art techniques. For extraction we use labeled supervised and semi-supervised data. For abstraction we use titles and subtitles for training the models.

4.1 Nomenclature and Base Model

Recurrent Neural Network is a type of neural network which is an extension to Feed Forward NN, with at least one feed-back connection, so activations can flow round in a loop [10, 36]. Essentially, information from prior observation along with the current observations are used to make predictions.

The notations are borrowed from Sutskever et al. [6]. RNN computes an output sequence $(y_1, y_2, \dots, y_T)$ for a given input sequence $(x_1, x_2, \dots, x_T)$ corresponding to the following equations:

$$h_t = sigm(W^{h_x} x_t + W^{h_h} h_{t-1}) \tag{2}$$

$$y_t = W^{y_h} h_t \tag{3}$$

This framework works when there is an alignment between input and output sequences that is the size of the input is same as size of the output. When the size of input and output sequences are different then two RNNs with encoding-decoding mechanisms are used (Cho et al. [6]). Theoretically this framework should work however it is found that RNNs with encoding-decoding mechanisms find difficulties in mapping long sequences or when there are long term dependencies. Gated Recurrent Units (GRUs) [8] and Long Short Term Memory (LSTMs)) [34] solve this problem by introducing gates into the network to prevent vanishing gradient problem associated with RNNs with long term dependencies.

In current study, LSTMs are used for extractive and abstractive summarizations. In standard LSTM sequence of fixed length is passed as an input to be encoded into a fixed dimension vector (v), which is then decoded into the output sequence of words. To summarize, LSTM estimates the following:

$$p(y_1, y_2, \ldots, y_{T'} | x_1, x_2, \ldots, x_T) = \prod_{t=1}^{T'} p(y_t | v, y_1, y_2, \ldots, y_{t-1})$$

(4)

In abstractive summarization, document can be fed as an input during training and the summaries can be fed as output. However, extractive RNN can be trained using standard supervised classification setup by performing soft-max on the encoded layer. [5].

4.2 Contextual Recurrent Neural Network

4.2.1 Abstractive Contextual RNN (AC-RNN). It is an RNN architecture wherein a document context vector as described in Section 3 is passed as an input at first time step along with the document sequence in the encoder. The idea is that if a pre-learned document-context-vector (v_d) is passed as an input at the beginning of the encoding stage, then the model not only converges faster but also learns the summaries corresponding to the document and not just the generic sequences. The basic idea is that if a reader is aware of the *title* of a document or *abstract* for a publication, then it provides a better understanding and a high-level interpretation of the document which makes the model to be able to provide more specific summaries corresponding to the documents. Therefore, document-context-vector (v_d) at the first time-step essentially changes the encoding vector.

An LSTM decoder with similar architecture mentioned in encoding have been used, however the input at time t=0 to the decoder is the vector obtained from encoder [6, 37]. Unlike other decoding mechanisms, where output at time t can be any word from the vocabulary, the output from the document vocabulary is considered during the time of prediction, making the inference faster.

Sutskever et al. [37] and [39] proposed beam-search to obtain the most likely sentence in the machine translation task. However, we perform a heuristics by using the trained model for re-ranking all the sentences (within the document) based on likelihood of a sentence being a summary sentence during the inference. Where likelihood of any sentence is defined as likelihood of decoding the sentence given the encoded input. So, the abstractive model is used for extraction during the inference. We adopted the idea from a state-of-the-art techniques for Automatic Speech Recognition, wherein an RNN is used for re-ranking the potential outcomes from an n-gram based language model [40]. We do this to address several issues: (a) Avoid generic and short output issues with sequence to sequence models, (b) Obtaining grammatically correct sentences for eBay users to avoid poor customer experience and (c) Avoiding legal push-backs from the sellers.

4.2.2 Extractive Contextual RNN (EC-RNN). It consists of Encoder only. The encoder used in EC-RNN is the replica of encoder used in AC-RNN with document-context-vector (v_d) as an input at time t=0 and embedding representation of the words is passed as input to the model. However, the output of encoder is used for binary classification (sentence being a summary sentence or a not) using softmax. Note that, each sentence starts with document-context-vector, therefore the classification of sentence happens given the context of the document and not just the sentence alone. In this way same sentence may be classified as summary sentence for

Table 1: Description distribution of a eBay products.

Total vocabulary size	768,298
Median document length	346 characters
Median number of words	54
Median sentence length	51 characters
Median Nbr of words in sentence	8

one document but not for others. Furthermore, given a document-context-vector (v_d), it is the extra information provided by the sentence which differentiates it from the other sentences in the document, which is a major drawback of other state-of-the-art classification approaches wherein some sentences are always classified as a summary sentence.

4.3 Non-Contextual RNN Architectures

In current setup, RNNs trained without document-context-vector are termed as non-contextual RNNs. Recently, several architectures have been proposed in this regards [1, 7, 15, 21, 28, 32], however three main models which are explored in current study are:

4.3.1 Abstractive RNN (A-RNN). Abstractive RNN is the traditional sequence to sequence model using LSTM suggested by [6, 37]. The model is exactly similar to AC-RNN without the context as input at time t=0. The input at time t=0 in A-RNN is a token *<start>*. A fixed sized input and output sequences are used for training by either curtailing or padding.

4.3.2 Extractive RNN (E-RNN). RNN has been used for classification tasks and it generates state-of-the art results. Extractive RNN is non-context version of EC-RNN proposed in Section 4.2.2. As mentioned before, embeddings for words are pre-calculated using Skip Gram with Negative Sampling (SGNS) technique and are used as input corresponding to each word in encoding layer. Features are extracted using embedding for classification task.

4.3.3 Convolutional RNN (CNN-RNN). Convolution based LSTM has performed extremely well in text classification tasks [42]. Furthermore, convolution attention-based encoder [7] has been used for short summarization tasks. CNN-LSTM is used to classify the sentences with the same technique as E-RNN, however the difference is that CNN is used to extract sequences of higher-level phrase representations. As suggested by Zhou et al. [42] CNN-LSTM is able to capture both local features of phrases, global and temporal sentence semantics. CNNs with multiple filters, max-pooling and dropout are used to extract high-level phrase representations and then passed to LSTM for classification using Softmax.

5 DATASETS

Table 1 describes the distribution of a eBay description. Vocabulary size of our dataset is 768K words. Median document length is 346 and 54 words.

5.1 Datasets

There are two kinds of datasets which are used in current study.

Human Extracted Snippets: 20,000 items/documents and corresponding details (titles, url, description) were provided to humans

for extracting the summaries. The task was to extract and rank the sentences from the descriptions given eBay item url. 5k items out of 20k items were used for evaluations (Golden Set) and 15,000 items were used for training different models.

Semi-supervised Large Scale Summarization Approximation: Sentences from 100,000 item-descriptions were extracted and ranked based on relevance towards the document context. Section 3.4 provides information about how to rank a sentence in the order of relevance given the document and it's contextual details. Several techniques have been proposed by Shen et al., Lin et al. and [2, 13] based on query, thematic similarity, topic signature and Latent Semantic Analysis . [29] expanded the topics to Wikipedia and obtained best results on DUC summarization task.

Similar to approaches mentioned above, we obtain the approximate summaries for eBay item descriptions. After an evaluation from a eBay reviewers, it is found that summaries generated using document-context based approach are of high quality and can be used for training the models. Given the quality of these summaries, eBay launched this feature on mobile applications and website. On an A/B test, it is found that showing summaries using this approach have a high monetary value when compared to not showing the summary-snippet. Models trained using semi-supervised approach will be evaluated on Golden test set to identify the relevance of this technique.

5.2 Data Generation for Classification task

EC-RNN, C-RNN, CNN-RNN and other classification based Extractive summarization techniques need the labeled data for training. For classification tasks, sentences which have blacklist terms are labeled as *non-summary* class while the sentences scoring high on document-context metric are considered as positive sentences. Blacklist terms are the words and phrases which do not contain item/document level information and are frequently used at eBay such as "returns", "shipping","rate me 5 stars", etc. We obtained 700 terms using human curation and with statistical analysis of eBay item descriptions. For each description in 100,000 items, sentences are tagged as positive or negative. Sentences which were neither scoring high on document-context metric nor having blacklist terms were left out from being tagged. Since, the document-context sentence ranking is a new approach and is yet to be evaluated except using A/B test in production, the data is tagged for high-precision. This work is a step towards evaluating document-context based sentence ranking as well.

6 ARCHITECTURE DETAILS AND EXPERIMENTATION

6.1 Training Details and Model Architectures

Abstractive Context RNN (AC-RNN) and Abstractive RNN (A-RNN) were trained using deep LSTM with 4 layers (as described in Sutskever et al. [37]) with 1000 cells and 300 dimension word embeddings. Since, we wanted to find the relative difference after adding the context in RNNs for summarization task, we kept the same parameters for both the models. Parameters settings and model details which worked the best in our case are mentioned in Table 2.

Table 2: Parameter setting for Abstractive approaches like AC-RNN and A-RNN.

Parameters	Value
Input description length	50 words
Output summary length	15 words
Optimization Method	Stochastic Gradient Descent with momentum
Learning rate	0.1; reduced to half after every third epoch.
Batch size	128
LSTM Parameters	Uniform distribution from [-0.1, 0.1]

Table 3: Parameter setting for Extractive approaches like EC-RNN and E-RNN.

Parameters	Value
Maximum sentence length	15 words
Optimization Method	Adam
Learning rate	0.01
Batch size	256
LSTM Parameters	Uniform distribution from [-0.1, 0.1]

Table 4: Parameter setting for Convolutional RNN .

Parameters	Value
Maximum sentence length	15 words
Dropout keep probability	0.5
Learning rate	0.01
Filter sizes for Convolution	4
Batch size	128
CNN and LSTM Parameters	Random Normal centered at 0 with standard deviation 0.1
Max pool size	4

Extractive Context RNN (EC-RNN) and Extractive RNN (E-RNN) were trained using two LSTM layers with 300 cells and 300-dimension word embedding. Since, we wanted to find the relative difference after adding the context in RNNs for summarization task, we kept the same parameters for both the models. Parameters settings and model details which worked the best in our case are mentioned in Table 3.

Convolutional RNN (CNN-RNN) consists of two neural networks. CNN for high level phrase representations and then LSTMs for obtaining the temporal and sequential nature of the text. We used single layer Convolution with filter size equal to 4 and a Single layer LSTM with 300 cells and 300-dimension word embeddings were used. Parameters settings and model details which worked the best in our case are mentioned in Table 4.

6.2 Experiments and Evaluation Metrics

We split our 20K human extracted summaries dataset into 15K and 5K parts. We use 5K for all evaluations. We also used the data obtained using semi-supervised technique for training and evaluating

Table 5: Different type of experiments done and metrics used for those experiments.

Evaluation Setting	Description	Evaluation Metric
Classification	Given a sentence, classify whether it is summary sentence or not.	Accuracy, Precision, Recall, F-score
Similarity	Given golden summaries, find the similarity score	ROUGE (ROUGE-1, ROUGE-2, ROUGE-L) [20], BLEU, TF-IDF Cosine Similarity, Topic Similarity
Ranking	Given sentences, rank them by in the order of relevance towards summary	NDCG, Mean Average Precision

Table 6: Results on supervised task using human extracted summaries - 15K for training and 5K for evaluation. 2 variations with target summary :- (a) 1 sentence long and (b) 3 sentences long. Abstractive Context RNN (AC-RNN) performs best in all metrics for short summaries. Extractive Context RNN (EC-RNN) performs best for longer summaries. Adding context as a whole created improvements in RNN models.

Model	Token Sim.	$Rouge_1$	$Rouge_2$	Rouge -LCS	BLEU	Topic Sim.
Target snippet length = 3 sentence						
Fuzzy	0.18	0.26	0.20	0.20	0.15	0.01
E-RNN	0.20	0.27	**0.37**	0.27	0.16	0.02
EC-RNN	**0.26**	**0.39**	0.22	**0.30**	**0.25**	0.02
CNN-RNN	0.21	0.29	0.21	0.27	0.21	0.02
A-RNN	0.19	0.27	0.28	**0.38**	0.19	0.02
AC-RNN	**0.22**	**0.33**	**0.31**	0.27	0.22	0.02
NB	0.19	0.28	0.25	0.24	0.15	0.01
SVM	0.20	0.27	0.22	0.26	0.19	0.01
LSA	0.20	0.30	0.24	0.30	0.17	0.02
LexRank	0.20	0.28	0.25	0.24	0.18	0.02
TextRank	0.20	0.31	0.23	0.28	**0.24**	0.02
Target snippet length = 1 sentence						
Fuzzy	0.11	0.15	0.18	0.25	0.09	0.003
E-RNN	0.12	0.15	**0.23**	0.28	0.10	0.004
EC-RNN	**0.13**	**0.17**	0.22	**0.29**	0.11	0.004
CNN-RNN	0.12	0.15	0.19	0.29	0.10	0.004
A-RNN	**0.13**	0.16	0.22	**0.29**	0.11	0.004
AC-RNN	**0.14**	**0.19**	**0.24**	**0.30**	**0.11**	**0.006**
NB	0.12	0.15	0.21	0.25	0.11	0.004
SVM	0.12	0.15	0.20	0.27	0.11	0.004
LSA	0.12	0.16	0.21	0.26	0.10	0.005
LexRank	0.12	0.15	0.19	0.25	0.10	0.004
TextRank	0.11	0.15	0.20	0.24	0.10	0.003

Table 7: Results on supervised task using human extracted summaries - 15K for training and 5K for evaluation. 2 variations with target summary: When target summaries are extremely long (5 sentence). Extractive Context RNN (EC-RNN) performs best. Adding context as a whole created improvements in RNN models.

Model	Token Sim.	$Rouge_1$	$Rouge_2$	Rouge -LCS	BLEU	Topic Sim.
Target snippet length = 5 sentence						
Fuzzy	0.28	0.37	0.28	0.21	0.30	0.03
V-RNN	0.33	0.41	**0.61**	0.28	0.29	0.03
C-RNN	**0.40**	**0.52**	0.40	0.31	**0.41**	0.04
CNN-RNN	0.32	0.43	0.29	0.26	0.36	0.04
A-RNN	0.31	0.36	0.29	**0.33**	0.37	0.03
AC-RNN	0.36	0.45	0.31	0.28	0.37	0.04
NB	0.31	0.40	0.42	0.28	0.18	0.03
SVM	0.32	0.40	0.38	0.29	0.25	0.03
LSA	0.30	0.41	0.40	0.28	0.21	0.03
LexRank	0.30	0.40	0.30	0.28	0.28	0.03
TextRank	0.31	0.43	0.39	0.27	0.31	0.03

Table 8: Classification for Extractive Supervised Model (Classes: 0 -> Non-summary). Training - 15K human judged, Evaluation-5K human judged. Extractive Context RNN (EC-RNN) shows best performance. Note: No classification for abstractive models.

Model	Accuracy	Precision (0,1)	Recall (0,1)	F-score (0,1)
NB	94.11	95, 92.32	96, 90	95.5, 91.4
SVM	93.31	97.13, 87.18	92.4, 94.91	94.74, 90.90
E-RNN	95.4	98.42, 90.37	94.47, 97.12	96.40, 93.62
CNN-RNN	96.01	96.22, 95.56	97.69, 92.81	96.94, 94.16
EC-RNN	**98.1**	**99.0, 96.07**	**97.85, 98.27**	**98.42, 97.16**

the models. **Supervised** - we train our models on 15K human extracted summaries. These models are trained with three different settings of target summary lengths. We use 1 sentence, 3 sentence and 5 sentence summary lengths.

Semi-supervised - we use 100K documents and approximate summaries generated via document context for training models at scale. We use 5K human extracted summaries for evaluation purposes. We also baseline with a fuzzy summarization strategy where random sentences are picked in the summary output.

6.3 Results for Supervised Setting

In this section we present the result for supervised models for summarization. For this puproses we trained our models on 15K human extracted summaries and evaluated on 5K human extracted summaries.

Table 6 compiles the performance result for all summarization strategies. It is clear that RNN's as a whole outperform other techniques (Naive Bayes, SVM, LSA, LexRank, TextRank) on Rouge, BLEU and token similarity. In both abstractive and extractive RNN's adding document-context improves all metrics. For example, Rouge-1 and BLEU for abstractive RNN is 0.27 and 0.19 respectively. When

Table 9: Supervised model ranking evaluation. Training - 15K human judged, Evaluation-5K human judged. Extractive Context RNN (EC-RNN) shows best performance.

Model	NDCG@1	NDCG@3	MAP@1	MAP@3
Fuzzy	0.535	0.576	0.087	0.144
E-RNN	0.617	0.636	0.094	0.159
EC-RNN	**0.642**	**0.682**	0.102	**0.162**
CNN-RNN	0.615	0.658	0.090	0.151
A-RNN	0.630	**0.679**	**0.109**	0.155
AC-RNN	0.635	0.667	**0.116**	**0.160**
NB	0.619	0.611	0.092	0.151
SVM	0.620	0.646	0.096	0.156
LSA	**0.639**	0.651	0.096	0.153
LexRank	0.607	0.644	0.095	0.153
TextRank	0.629	0.645	0.095	0.153

Table 10: Classification for Semi-supervised Extractive Supervised model (Classes: 0-> Non-summary). Training - 100K algorithmically labeled data, Evaluation-5K human judged. EC-RNN shows best performance.

Model	Accuracy	Precision (0,1)	Recall (0,1)	F-score (0,1)
Semi-supervised				
E-RNN	96.77	99.37, 92.39	95.64, 98.86	97.47, 95.52
CNN-RNN	97.9	97.36, 98.93	99.41, 95.22	98.38, 97.04
EC-RNN	**99.41**	**99.94, 98.43**	**99.14, 99.90**	**99.54, 99.16**
Supervised				
E-RNN	95.4	98.42, 90.37	94.47, 97.12	96.40, 93.62
CNN-RNN	96.01	96.22, 95.56	97.69, 92.81	96.94, 94.16
EC-RNN	**98.1**	**99.0, 96.07**	**97.85, 98.27**	**98.42, 97.16**

document context is added (AC-RNN) then Rouge-1 and BLEU increases to 0.33 and 0.22 respectively. Similarly for Extractive RNN (E-RNN) vs extractive contextual RNN (EC-RNN) rouge-1 changes from 0.27 to 0.39 and BLEU changes from 0.16 to 0.25.

For large target summaries which are 3 sentence long *extractive contextual RNNs* perform the best followed by *abstractive contextual RNNs*. For example rouge-1 for EC-RNN is 0.39 and for AC-RNN is 0.33. Similarly BLEU is 0.25 compared to 0.22 for AC-RNN. For small target summaries (1 sentence long) we observe that abstractive-context-rnn (AC-RNN) outperform extractive-context-rnn (EC-RNN). Where rouge-1 is 0.15 for EC-RNN compared to 0.19 for AC-RNN. For target summaries which are 5 sentences long results are shared in Table 7. As the target summary length increases efficacy of extractive contextual RNN increases over abstractive contextual RNNs.

In Table 8 we present classification metrics for extractive models. All models are doing a good job of separating summary-sentences from non-summary sentences. For this task EC-RNN outperforms other methodologies as well.

Table 9 describe ranking metrics for summarization models. We assign summary sentences as relevance of 1 and other sentences relevance score of 0. The task is to generate ranking of sentences in a way that picks the summary sentences before non-summary sentences. For this task as well context aware RNNs win out. Extractive context RNN (EC-RNN) have the highest NDCG@1, NDCG@3 and MAP@3.

6.4 Results for Semi-supervised Setting

In this section we share the result of training with algorithmically generated approximate summaries. These approximate summaries are generated by using document-context. For extractive RNN and other classification approaches like SVM, NB human labeled and algorithmically labeled data was used for training. Whereas for abstractive RNN we use title and subtitles for learning.

Table 10 compares the classification metric for extractive models when trained with human extracted summaries to algorithmically labeled approximate summaries. It can be seen that training models on large scale approximate summaries does not lead to any drop in precision-recall and accuracy.

Table 11 compares the summarization results for all RNN models (abstractive vs extractive). Training on 100K approximate summaries does not lead to drop in metric. For example Rouge-1 drops from 0.39 (EC-RNN supervised) to 0.38 (EC-RNN semi-supervised). BLEU also remains comparable with 0.26 for 100K-dataset and 0.25 for 15-K dataset.

Ranking metrics are compared in Table 12. As the size of data increases for training abstractive methods performance increases tremendously. For example AC-RNN have NDCG@3 of 0.667 vs 0.715 for 15K vs 100K documents respectively. Same result hold for MAP as well, where MAP@3 increase from 0.16 to 0.17 when data is increased from 15K to 100K for AC-RNN. For extractive RNN, which are trained on approximate summaries we don't see a drop in ranking metrics between 15K human extracted vs 100K approximate summaries. MAP@3 remains 0.16 in both cases.

Overall it can be seen that abstractive RNNs are improving with more data. Extractive RNN are able to generate near similar performance with large-scale approximate summaries as with small scale human extracted summaries. And adding document-context to RNN with approximate summaries further boost the performance for both abstractive and extractive RNN.

7 CONCLUSION

We proposed a novel Document-context Seq2Seq model for abstractive and extractive summarization. We have shown that RNNs and other Seq2Seq models are powerful and beat state-of-the-art summarization approaches in e-Commerce setting. The idea of adding contextual information at the first time-step during the encoding of the input to output sequence/label mapping aligns with humans since generally humans tend to read title, abstract and gather other contextual information before reading the entire document/articles. This gives humans high-level understanding of the document, which if incorporated in Seq2Seq model will generate much richer document specific summaries. Training is performed in a human tagged supervised setting as well as with Large-Scale-Semi-Supervised extracted summaries. It is found that Seq2Seq based RNN summarization techniques out-performs other state-of-the-art summarization techniques. Within RNNs, Contextual-RNNs outperform Non-Contextual-RNNs on most of the similarity and ranking

Table 11: Result of summarization for Semi-supervised Extractive Supervised Model. Training - 100K algorithmically labeled data, Evaluation-5K human judged. Extractive Context RNN (EC-RNN) shows best performance. Abstractive Contextual RNNs show significant improvements with large training data.

Model	Token Sim.	$Rouge_1$	$Rouge_2$	Rouge -LCS	BLEU	Topic Sim.
Semi-supervised						
E-RNN	0.21	0.28	**0.37**	0.28	0.16	0.015
EC-RNN	**0.27**	**0.38**	0.23	0.30	**0.26**	**0.023**
CNN-RNN	0.21	0.30	0.21	0.28	0.22	0.018
A-RNN	0.21	0.27	0.29	**0.41**	0.21	0.016
AC-RNN	0.23	0.33	0.32	0.29	0.22	0.021
Supervised						
E-RNN	0.20	0.27	**0.37**	0.27	0.16	0.02
EC-RNN	**0.26**	**0.39**	0.22	0.30	**0.25**	0.02
CNN-RNN	0.21	0.29	0.21	0.27	0.21	0.02
A-RNN	0.19	0.27	0.28	**0.38**	0.19	0.02
AC-RNN	0.22	0.33	0.31	0.27	0.22	0.02

Table 12: Ranking metrics for Semi-supervised Extractive Supervised model. Training - 100K algorithmically labeled data, Evaluation-5K human judged. Abstractive contextual RNNs show most improvements with large training data. Extractive contextual RNNs also show improvements.

Model	NDCG@1	NDCG@3	MAP@1	MAP@3
Semi-supervised, Supervised				
E-RNN	0.625, 0.617	0.659, 0.636	0.097, 0.094	0.158, 0.159
EC-RNN	0.655, **0.642**	0.708, **0.682**	0.103, 0.102	0.167, **0.162**
CNN-RNN	0.622, 0.615	0.676, 0.658	0.090, 0.090	0.154, 0.151
A-RNN	0.649, 0.630	0.674, 0.679	0.116, 0.109	0.160, 0.155
AC-RNN	**0.677**, 0.635	**0.715**, 0.667	**0.125**, **0.116**	**0.170**, 0.160

measures. Extractive-Contextual RNNs are found to be best performing followed by Abstractive-Contextual RNNs for large summaries, however for shorter summaries Abstractive-Contextual-RNNs outperform all other techniques followed by Extractive-RNNs, with Attention and more sophisticated setting it can be possible to further improve Abstractive techniques. We have also depicted that Abstractive-RNNS (Contextual/Non-Contextual) can be used for Extraction tasks and still beat the Extractive systems. It is found that Large-Scale-Semi-Supervised data for training improves the performance of the models on the golden-evaluation-dataset. Hence, training the extractive models with approximate summaries leads to better results compared to relatively smaller human tagged supervised data. We think that advantage of large scale training outperforms the noise in approximating summaries. We recommend other researchers to incorporate context in other Seq2Seq tasks (e.g. machine translation task).

REFERENCES

[1] Text summarization with tensorflow. https://ai.googleblog.com/2016/08/text-summarization-with-tensorflow.html. Accessed: 2017-10-23.

[2] R. M. Badry, A. S. Eldin, and D. S. Elzanfally. Text summarization within the latent semantic analysis framework: comparative study. *International Journal of Computer Applications*, 81(11), 2013.

[3] R. Barzilay and M. Elhadad. Using lexical chains for text summarization. *Advances in automatic text summarization*, pages 111–121, 1999.

[4] T. Baumel, M. Eyal, and M. Elhadad. Query focused abstractive summarization: Incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models. *arXiv preprint arXiv:1801.07704*, 2017.

[5] J. Cheng and M. Lapata. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*, 2016.

[6] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[7] S. Chopra, M. Auli, A. M. Rush, and S. Harvard. Abstractive sentence summarization with attentive recurrent neural networks. *Proceedings of NAACL-HLT16*, pages 93–98, 2016.

[8] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[9] D. Das and A. F. Martins. A survey on automatic text summarization. *Literature Survey for the Language and Statistics II course at CMU*, 4:192–195, 2007.

[10] J. L. Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

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

[12] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. Dauphin. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*, 2017.

[13] Y. Gong and X. Liu. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–25. ACM, 2001.

[14] F. Guo, A. Metallinou, C. Khatri, et al. Topic-based evaluation for conversational bots. *arXiv preprint arXiv:1801.03622*, 2017.

[15] U. Khandelwal. Neural text summarization.

[16] C. Khatri, N. Parikh, S. Solanki, et al. Snippet extractor: recurrent neural networks for text summarization at industry scale, 2016. US Patent 15,237,091.

[17] C. Khatri, N. Parikh, S. Solanki, et al. Snippet generation and item description summarizer, 2016. US Patent 15,267,346.

[18] C. Khatri, S. Voleti, S. Veeraraghavan, N. Parikh, A. Islam, S. Mahmood, N. Garg, and V. Singh. Algorithmic content generation for products. In *Big Data (Big Data), 2015 IEEE International Conference on*, pages 2945–2947. IEEE, 2015.

[19] P. Li, W. Lam, L. Bing, and Z. Wang. Deep recurrent generative decoder for abstractive text summarization. *arXiv preprint arXiv:1708.00625*, 2017.

[20] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain, 2004.

[21] K. Lopyrev. Generating news headlines with recurrent neural networks. *arXiv preprint arXiv:1512.01712*, 2015.

[22] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.

[23] R. Mihalcea and P. Tarau. Textrank: Bringing order into texts. Association for Computational Linguistics, 2004.

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[25] G. A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, Nov. 1995.

[26] I. F. Moawad and M. Aref. Semantic graph reduction approach for abstractive text summarization. In *Computer Engineering & Systems (ICCES), 2012 Seventh International Conference on*, pages 132–138. IEEE, 2012.

[27] R. Nallapati, F. Zhai, and Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. *arXiv preprint arXiv:1611.04230*, 2016.

[28] R. Nallapati, B. Zhou, C. Gulcehre, B. Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

[29] V. Nastase. Topic-driven multi-document summarization with encyclopedic knowledge and spreading activation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 763–772. Association for Computational Linguistics, 2008.

[30] R. Paulus, C. Xiong, and R. Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.

[31] A. Ram, R. Prasad, C. Khatri, and A. Venkatesh. Conversational ai: The science behind the alexa prize. *arXiv preprint arXiv:1801.03604*, 2017.

[32] A. M. Rush, S. Chopra, and J. Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

[33] G. Singh, N. Parikh, and N. Sundaresn. User behavior in zero-recall ecommerce queries. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 75–84. ACM, 2011.

[34] H. F. Song, G. R. Yang, and X.-J. Wang. Training excitatory-inhibitory recurrent neural networks for cognitive tasks: A simple and flexible framework. *PLoS*

Comput Biol, 12(2):e1004792, 2016.

[35] J. Steinberger and K. Jezek. Evaluation measures for text summarization. *Computing and Informatics*, 28(2):251–275, 2009.

[36] I. Sutskever, J. Martens, and G. E. Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024, 2011.

[37] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[38] A. Venkatesh, C. Khatri, A. Ram, F. Guo, et al. On evaluating and comparing conversational agents. *arXiv preprint arXiv:1801.03625*, 2017.

[39] S. Wiseman and A. M. Rush. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960*, 2016.

[40] W. Xiong, Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke. The microsoft 2017 conversational speech recognition system. *arXiv preprint arXiv:1708.06073*, 2017.

[41] Y. Xu, J. Lau, T. Baldwin, and T. Cohn. Decoupling encoder and decoder networks for abstractive document summarization. *aclweb DOI10.18653/v1/W17-1002*, 2017.

[42] C. Zhou, C. Sun, Z. Liu, and F. Lau. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630*, 2015.