Discriminative Pre-training for Low Resource Title Compression in Conversational Grocery

Snehasish Mukherjee
Walmart Labs
Sunnyvale, California
smukherjee@walmartlabs.com

Phaniram Sayapaneni
Walmart Labs
Sunnyvale, California
phaniram.sayapaneni@walmartlabs.com

Shankar Subramanya
Walmart Labs
Sunnyvale, California
ssubramanya@walmartlabs.com

ABSTRACT

The ubiquity of smart voice assistants has made conversational shopping commonplace. This is especially true for low consideration segments like grocery. A central problem in conversational grocery is the automatic generation of short product titles that can be read out fast during a conversation. Several supervised models have been proposed in the literature that leverage manually labeled datasets and additional product features to generate short titles automatically. However, obtaining large amounts of labeled data is expensive and most grocery item pages are not as feature-rich as other categories. To address this problem we propose a pre-training based solution that makes use of unlabeled data to learn contextual product representations which can then be fine-tuned to obtain better title compression even in a low resource setting. We use a self-attentive BiLSTM encoder network with a time distributed softmax layer for the title compression task. We overcome the vocabulary mismatch problem by using a hybrid embedding layer that combines pre-trained word embeddings with trainable character level convolutions. We pre-train this network as a discriminator on a replaced-token detection task over a large number of unlabeled grocery product titles. Finally, we fine tune this network, without any modifications, with a small labeled dataset for the title compression task. Experiments on Walmart's online grocery catalog show our model achieves performance comparable to state-of-the-art models like BERT and XLNet. When fine tuned on all of the available training data our model attains an F1 score of 0.8558 which lags the best performing model, BERT-Base, by 2.78% and XLNet by 0.28% only, while using 55 times lesser parameters than both. Further, when allowed to fine tune on 5% of the training data only, our model outperforms BERT-Base by 24.3% in F1 score.

CCS CONCEPTS

• **Information systems** → **Summarization**; • **Human-centered computing** → *Natural language interfaces*; • **Computing methodologies** → *Natural language processing*; *Learning latent representations.*

KEYWORDS

title summarization, pre training, replaced token detection, representation learning, low resource summarization

SIGIR eCom'20, July 30, 2020, Virtual Event, China

ACM Reference Format:
Snehasish Mukherjee, Phaniram Sayapaneni, and Shankar Subramanya. 2020. Discriminative Pre-training for Low Resource Title Compression in Conversational Grocery. In *Proceedings of ACM SIGIR Workshop on eCommerce (SIGIR eCom'20).* ACM, New York, NY, USA, 7 pages.

1 INTRODUCTION

Voice commerce is a rapidly growing vertical, with an estimated market size of $2 Billion already and forecasts for hitting $40 Billion+ by 2022 [27]. Moreover, voice is being touted as the dominant channel for driving customer engagement in the next decade. As such, conversational commerce, a broader term often used interchangeably with voice commerce, is an area of strategic priority for most e-commerce businesses. From a user adoption and repeat usage standpoint, the prospects for voice commerce are even more exciting for low consideration segments like Grocery.

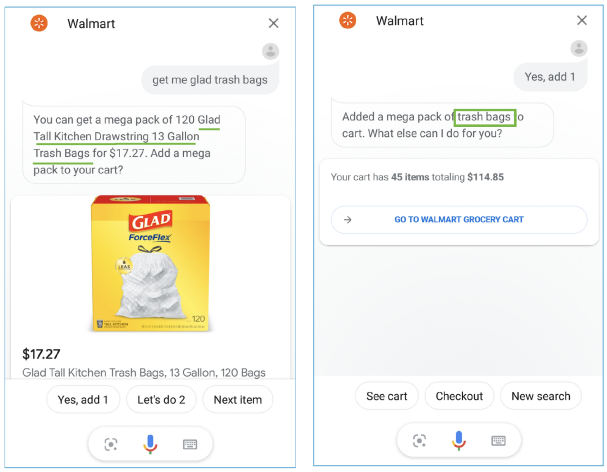

Figure 1: Walmart Voice Order bot for online grocery. Though search response is fairly detailed, the add to cart response has the maximum possible compression of the product title.

This big potential notwithstanding, the user experience for voice shopping, and conversational grocery in particular, is still nascent. One such suboptimality is the reading out of long product titles during a conversation between an artificial shopping agent and a customer. As noted in [22], product titles in e-commerce catalogs are made very long and informative with Search Engine Optimization

in mind. However, this translates to a poor grocery voice shopping experience where transactions are mostly repeat purchases and users are well aware of the products that are being added to cart. Reading out a product title like "Glad OdorShield Tall Kitchen Drawstring Trash bags - Febreze Fresh Clean - 13 Gallon - 40 count (Packaging May Vary)" for 7 to 10 seconds during an add-to-cart response, for example, is an unnecessary test of user's patience and can lead to greater drop out rates before checkout. Instead, a succinct summarization of the title, like "trash bags", is more appropriate in this scenario. From a voice UX standpoint, any product title with >= 5 words is considered to be long, while those with <= 4 words are considered short and voice friendly. This necessitates exploring automatic product title summarization techniques that can take in a long product title and convert it into a short and voice friendly title.

Automatic generation of short product titles have been studied in the context of voice and mobile shopping in [12, 22, 24, 27, 28]. All proposed methods are supervised learning models that leverage labeled training data and other product features to compress titles. Much better results can be obtained, if this labeled data is used to fine tune state-of-the-art pre-trained models like BERT, XLNet, ELECTRA [5] etc. However, this poses a few problems in the context of conversational grocery. Firstly, labeled data is expensive to obtain and grocery product pages and catalog are not as feature-rich as some other segments like fashion and furniture, for example. Secondly, heavily parameterized models like BERT and XLNet that require more powerful GPUs to train, do not make a strong case in favor of cost-effectiveness, which is a carefully tracked metric for low margin segments like grocery.

We try to address these challenges in this paper by proposing a lightweight self-attentive BiLSTM architecture that can be pre-trained on unlabeled long product titles to obtain performance comparable to state-of-the-art models for the title summarization task. Since we are low on training data, we adopt the discriminative pre-training strategy, recently introduced in [5], for our model. Pre-training models as discriminators on a replaced token detection task results in higher sample efficiency and has been shown to outperform masked language model based pre-training. Experiments on Walmart's online grocery catalog show that our pre-trained model achieves performance comparable to BERT, XLNet, and ELECTRA while using 55X and 7X lesser parameters respectively. Further, when allowed to fine tune on less than 20% of the available training data, our pre-trained model outperforms all three.

2 RELATED WORK

2.1 Text Summarization

Our problem can be classified under the broad category of text summarization. This area has evolved quite a bit with the advent of RNNs. [1, 6, 11, 18] present comprehensive surveys of neural as well as classical text summarization techniques. While document level summarization [2, 3, 16] deals with the problem of generating document level summaries of the content, our work is best described as sentence compression [4, 8, 9, 15, 23, 25, 29] which involves summarizing long sentences to shorter ones while preserving the core intent.

There are 2 distinct flavors of summarization independent of the source granularity; *Abstractive* and *Extractive*. Abstractive summarization [2, 4, 17, 20] can produce summaries consisting of words not present in the input, while Extractive summarization [8, 9, 23, 29] aims to generate summaries using words or sentences extracted from the original input. We will focus on neural approaches for extractive flavor of sentence compression for the remainder of this section.

A popular approach to extractive sentence compression is to model the problem as a sequence-to-sequence learning problem using an encoder-decoder based architecture [8, 9, 15]. The encoder learns a distributed representation of the input sentence which is then fed to the decoder which is trained to produce a binary 0/1 label for each input word denoting whether to delete or keep that word in the output summary. Beam search can be used [9, 25, 26] to sample from the decoder output probability distribution to generate the most likely compression.

[23] and [29] present 2 approaches different from the encoder-decoder based approaches. [29] uses Reinforcement Learning with KEEP and DELETE policies. Words are kept or deleted based on the policy while a pre trained language model provides feedback on the generated summary. This results in the system learning the optimal policy over time. Like all the other deletion based compression models, [23] still tags words in the input sequence with a 0/1 label, but treats it as a sequence labeling problem, instead of a sequence generation problem. Hence it does away with the decoder and employs stacked BiLSTM layers with a final CRF layer at the output that does the 0/1 classification. Our current work closely follows this approach with minor modifications.

Most of the techniques discussed here, with the exception of [23], require a considerable amount of training data. While [8, 15, 29] are trained on the Gigaword corpus, [9] is trained on 2 Million news headline and summary pairs. Though these 2 Million examples were synthetically generated following the method in [10], it is based on the syntactic structure of the English language and its parse trees. [8] also proposes an unsupervised approach involving generation of training data by adding noise, but the results lag behind supervised approaches. Since the distribution of the words in the product titles, and in general the vocabulary of our problem domain, is different from general English, data sparsity and unavailability of e-commerce specific embeddings pose a challenge for us. Interestingly, [23] reported good performance with only 10000 pairs of original and compressed sentences, which is the reason we chose to implement a modified version of this approach to solve our problem of product title compression.

2.2 Title Summarization in E-commerce

Product title summarization in the context of voice and mobile shopping has been studied in [12, 22, 24, 27, 28]. Amongst them, [12] defined title summarization as a sequence classification problem, where a binary decision is made at each word, given a long product title. A feature vector has been applied, comprising term frequency (tf) and inverse document frequency (idf) for each word. However, in this approach, large training data (500,000 samples) has been used and the results didn't prove to be significantly better than a simple BiLSTM approach. Another interesting approach has been

Table 1: Crowd Generated Short Titles

No.	Long Title	Short Title
1.	Freshness Guaranteed Sliced Fruit Cake, 13 oz	Fruit Cake
2.	OGX Hydrating + Teatree Mint Conditioner Salon, 25.4oz	Conditioner
3.	Child of Mine by Carter's Places and Spaces 3 Pocket Duffle Diaper Bag Gray	Diaper Bag
4.	Mainstays Stall Size 54" x 78" Medium Weight PEVA Shower Liner, 1 Each	Shower Liner

Table 2: Dataset Statistics

Metric	Value
No. of pre-training samples	256,298
Shortest/Median/Longest sample lengths	3/10/35 words
Word Vocabulary size	67,634
Character Vocabulary size	69
Words missing from embedding (UNK)	17041 (25.2%)
No. of title compression samples	40,445
Shortest/Median/Longest short titles	1/2/5 tokens
Shortest/Median/Longest long titles	4/10/35 tokens

proposed, where the problem has been defined as multi-task learning objective [24] to compress the product title using user search log data. The multi-task learning objective involves two networks, one network to select the most informative words from the product title as a compressed title and the other network to generate the user search query from the product title. These two networks have been constrained to share the same product title encoder and additionally, the attention distributions from these two networks were constrained to agree with each other. However, in this approach, the size of the training data is large (185,386 samples) and additionally requires user search data. Another recent approach has been proposed in which the product title summarization has been framed as a Binary Named Entity Recognition problem [27]. The model architecture involves a simple bi-directional LSTM encoder/decoder network (2 layer LSTMs) with an attention mechanism. ANOVA and post-hoc tests showed that with this approach, there was no statistical difference between model outputs and human-labeled short titles.

3 DATASET

Our main dataset consists of product titles, and their corresponding human generated summaries for **40,445** top selling Walmart grocery products during the calendar year 2018. Crowd workers were given the product titles and asked to generate their short summaries by choosing words to retain from the original title. Product description, brand name and category information were also provided to help workers decide on the most salient tokens for unfamiliar items. The task was intentionally vague about the target length of the summary so as to not introduce any bias. Examples provided were minimal identity preserving compression, with extra terms added rarely to improve fluency. Table 1 lists some samples from this dataset. We call this dataset the **title summary dataset**. Unlike [27] our short titles do not necessarily contain additional entities like brand, size, pack descriptors etc that can be easily obtained from structured catalog data.

We use an additional unlabeled dataset of around **256,298** long product titles of items published in Walmart's online grocery catalog, which is used for pre training the network. We refer to this dataset as the **product titles dataset**. Table 2 lists the statistics for both these datasets. The length distribution across the crowd generated summaries and the original product titles conforms well to the UX requirement which considers product titles with >= 5 words as long and those with <= 4 words as short and voice friendly.

4 MODEL

We model the problem as a binary sequence labelling problem, where each element of the input sequence is assigned the label **0** or **1**. The sequence elements labeled **1**, taken in order, forms the short title. For the sequence labelling problem we use a 3 layer architecture consisting of an embedding layer, an encoder layer, and a final point-wise classification layer.

Embedding layer. A major problem in applying pre-trained embeddings to a specific domain like retail is vocabulary mismatch where many private labels, brands, pack descriptors etc are treated as unknown words. To overcome this we use a combination of fixed pre-trained word embeddings and randomly initialized, trainable, character level embeddings as described in [19]. Hence, our model takes 2 inputs; $\mathbf{x}_w \in \mathbb{Z}^N$ which is a vector containing indices of words in the input product title, and $\mathbf{x}_c \in \mathbb{Z}^{N \times C}$ which contains the indices of the characters in each word, where N is the maximum sequence length and C is the maximum word length. We use character level convolutions (CharCNN) [14] on $\mathbf{x}_c$ to combine and project the character level embeddings for each word onto $\mathbb{R}^{e_{char}}$. We combine these two word embeddings using a highway network [21] to obtain the final embedding $\mathbf{x}_{emb}$.

$$\mathbf{x}^i_{wemb} = \text{word-embedding } (\mathbf{x}^i_w), \in \mathbb{R}^{e_{word}} \quad (1)$$

$$\mathbf{x}^i_{cemb} = \text{CharCNN } (\mathbf{x}^i_c), \in \mathbb{R}^{e_{char}} \quad (2)$$

$$\mathbf{x}^i_{emb} = \text{highway } ([\mathbf{x}^i_{cemb}; \mathbf{x}^i_{wemb}]), \in \mathbb{R}^{e_{char}+e_{word}} \quad (3)$$

Instead of repeating the details of how these components work we direct the reader to [14, 19, 21] for further details.

Encoder layer. The encoder layer uses 3 stacked layers of bidirectional LSTMs to obtain contextualized representation $\mathbf{x}^i_b \in \mathbb{R}^{2h}$ for the i_{th} sequence element as the concatenation of the hidden states, each of dimension h, from the forward and backward passes of the LSTM units in the 3rd layer

$$\mathbf{x}^i_b = \left[\mathbf{h}^{(i)[3]}_f; \mathbf{h}^{(i)[3]}_b\right], i \in \{1, 2, ...N\} \quad (4)$$

We further augment this contextualized representation $\mathbf{x}^i_b$ of each sequence element by using multiplicative self attention to jointly attend to all other sequence elements $\mathbf{x}^j_b, j \in \{1, 2, ...N\}$ without having to go through any gating mechanism. We thus obtain the final encoding $\mathbf{x}^i_{enc}$ for each sequence element as follows

$$\mathbf{e}_{ij} = \mathbf{x}_b^{i\top} \mathbf{W}_s \mathbf{x}_b^j, \quad \mathbf{e}_{ij} \in \mathbb{R} \tag{5}$$

$$\alpha_{ij} = \frac{\exp\left(\mathbf{e}_{ij}\right)}{\sum_{k=1}^{N} \exp\left(\mathbf{e}_{ik}\right)} \tag{6}$$

$$\mathbf{x}_{enc}^i = \sum_{j=1}^{N} \alpha_{ij} \mathbf{x}_b^i \tag{7}$$

where $\mathbf{W}_s \in \mathbb{R}^{2h \times 2h}$ is a trainable parameter matrix and α_{ij} determines the contribution of the j_{th} sequence element in computing the representation for the i_{th} sequence element.

Classification layer. In the final layer we project the contextualized embeddings for each sequence element as obtained from the encoder layer to $\mathbb{R}^2$ using a point-wise fully connected layer, parameterized by the weight matrix $\mathbf{W_c} \in \mathbb{R}^{2h \times 2}$ and the bias $\mathbf{b_c} \in \mathbb{R}^2$, which when operated upon by a softmax operator yields $\mathbf{y}_i$, the probability distribution across the output class labels for the i_{th} sequence element. Succinctly,

$$\mathbf{y}_i = \text{softmax}\left(\mathbf{W}_c^T \mathbf{x}_{enc}^i + \mathbf{b_c}\right), i \in \{1, 2, ...N\} \tag{8}$$

Training. We train our model to minimize the weighted binary cross entropy loss $L(\theta)$ given by

$$L(\theta) = -\frac{1}{N} \sum_{i}^{N} \alpha \cdot \hat{\mathbf{y}}_i \log\left(\mathbf{y}_i\right) + \beta \cdot (1 - \hat{\mathbf{y}}_i) \log(1 - \mathbf{y}_i) \tag{9}$$

where N is the sequence length, $\mathbf{y}_i$ is the probability that the i_{th} sequence element belongs to class **1**, $\hat{\mathbf{y}}_i$ is the ground truth label, α is the weight for class **0**, and $\beta = 1 - \alpha$, is the weight for class **1**. We choose $\alpha = 0.1$ and hence $\beta = 0.9$ since roughly $\frac{9}{10}$ of all the token labels are **0**

Our model architecture is similar to [9, 23, 27] with some key differences. Firstly, unlike our hybrid embedding layer, none of the aforementioned solutions have any mechanism to address the vocabulary mismatch problem between pre-trained embeddings and the training corpus. Next, unlike [23, 27] that use encoder-decoder based architecture, ours in an encoder-only architecture. Further, [9] uses left to right LSTM only and hence requires feeding the input in reverse to condition on the right context, which we achieve using BiLSTM. Finally, unlike [9, 23, 27] we attend to the global context while encoding each sequence position using a self attention layer.

5 PRE-TRAINING

As in [5], we pre-train our network as a discriminator on a replaced token detection task. We corrupt each long title in the product titles dataset, by randomly selecting some fraction f of its tokens and replacing them with another token. Though experiments in [7] show that best results are obtained with 15% of the tokens corrupted, we choose $f = 0.25$ since our corpus is much smaller. To ensure that the network gets a chance to make predictions for all positions, we repeat the token replacement process multiple times for the same long title until the corruption procedure covers all tokens. This results in multiple copies of the same title, with tokens replaced in mutually disjoint positions. We generate a binary sequence label for each corrupted line that has **1** for the positions that were replaced and **0** everywhere else. In order to not bias the network towards predicting at least one **1** label in each input, we also include the original, uncorrupted copy of each product title. This procedure applied on our dataset results in a corpus of 1.27 Million product titles for supervised pre-training of the network. We use the same weighted binary cross entropy loss function as in equation 9. Since the median number of tokens per title in our corpus is around 10, with $f = 0.25$ and $N = 35$ we used $\alpha = \frac{10f}{N} \approx 0.07$ and $\beta = 1 - \alpha = 0.93$. We train this network for 4 epochs which results in accuracy of 0.9897 on the replaced token detection task.

5.1 Token Replacement

Intuitively, higher the quality of the replaced token, harder it is for the network to guess that it is replaced, and hence better is the latent representations learnt during the pre-training process. Unlike [5] which samples the replacement tokens from a small masked language model trained jointly, but not adversarially, with the network, we obtain our replacement tokens from a skip gram model that maximizes the log likelihood in a window centered on the token to be replaced. This allows us to improve our compute efficiency while obtaining reasonably good candidate replacement tokens. If w_i is the token at the i_{th} position that is to be replaced, then we choose w_r, the replacement token as

$$w_r = \underset{w \in V'}{\text{argmin}} \sum_{k=-n}^{n} -\log P_s(w | w_{i+k}; l_w)$$

where V is the vocabulary, $V' = V - \{w_{i+k}; -n \leq k \leq +n\}$, $l_w = 2n + 1$ is the window size, and $P_s(w_i | w_j; l_w)$ is the conditional distribution for occurrence of w_i in a window of length l_w centered on w_j.

6 EXPERIMENTS

We implement our model and it's various ablations in the Tensorflow framework[1]. For our baseline models, we use their Pytorch implementations available at HuggingFace[2]. We minimize the model loss as given in equation 9 using the Adam optimizer. We do not tune any hyper parameters and use the default settings for Adam throughout with $lr = 0.001$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. We use the same dropout probability of 0.2 between all layers. All layer weights, with the exception of the highway layer and the character embeddings, use Xavier normal initialization and biases are set to 0. All our models are trained on a single Nvidia V100 GPU for 15 epochs, or 1 hour, or until convergence, whichever is earlier. We define convergence as 3 consecutive epochs without any improvements in metrics on the validation set. For fine-tuning our pre-trained model, we use gradual unfreezing of layers in a top down manner as in [13], but we get the best results when we keep our learning rate fixed at 0.001 across all layers and batches. We evaluate our model performance on a held out test set consisting of 8089 human generated short titles. We report 2 different metrics for our models; ROUGUE-1 F1 score and Exact Match percentage, henceforth referred to as **F1** and **EM** respectively. EM, as the name suggests,

[1] https://www.tensorflow.org/
[2] https://huggingface.co/transformers/

Table 3: Ablations

Model	F1	EM
CB3SA	0.8465	62.24
CB3SA+PT	**0.8558**	**63.83**
CB3SA-CharCNN	0.8414	60.13
CB3SA-BLSTM1	0.8455	62.37
CB3SA-SA	0.8417	60.22
CB3SA-SA+NWSA7	0.8458	62.39
CB3SA-SA+MHSA8	0.8420	59.72

refers to the percentage of outputs that exactly match the human generated title compression. For comparing model performances we use the **F1** score only.

6.1 Dataset Preparation

We minimally normalize our datasets by converting everything to lower case and squeezing consecutive white space characters. Given the nature of the Walmart catalog data, 2 additional normalization steps were necessary; converting all "&" symbols to "and" and padding commas with whitespaces so that they are treated as separate tokens. We follow a simple tokenization scheme where we split each product title by whitespace. We keep the maximum sequence length at 35, truncating longer product titles and padding shorter titles with the PAD token. The maximum token length we use is 15, with similar padding and truncating scheme. We extract our word and character vocabularies from the product titles dataset instead of the title summary dataset since the former is a super set of the later. Our corpus has a word vocabulary size of 67634 including the "UNK" and "PAD" tokens, while the character vocabulary size stands at 69. We set aside 20% of the human generated title summary dataset as our test set, 8% as validation set and the remaining 72% as our training set.

6.2 Ablations

We perform several ablations, as well as some additions, on the model proposed in section 4 to underscore the effect of each of the components. We name our model CB3SA, choosing to retain the first character for each layer (CharCNN, BiLSTM(3 stacks) and self attention). Additions and ablations are denoted with a + and - operator respectively after the model name. For example CB3SA+PT refers to the pre-trained version of our model, while CB3SA-CharCNN refers to the version of our model with the character level word embedding layer removed. Table 3 lists the various combinations we tried and the results on the test set.

Clearly, the pre-training process (model CB3SA+PT) contributed a significant boost in performance over the non pretrained model (CB3SA) while removing the charCNN layer (CB3SA-CharCNN) causes the most significant drop in performance. Apart from these, there are a few interesting observations. Firstly, removing a BiLSTM layer (model CB3SA-BLSTM1) has one of the least negative impacts together with reduced training and inference times. Secondly, removing the vanilla self attention layer and adding a multi-headed self attention layer with 8 attention heads (model CB3SA-SA+MHSA8) proves to be as detrimental as removing the self attention layer altogether (model CB3SA-SA). Finally, using a narrow width attention with window length of 7 instead of our global self attention (model CB3SA-SA+NWSA7) causes the least drop in performance.

Table 4: Comparison against SOTA models

Model	Params	F1	EM
CB3SA+PT	**2M**	**0.8558**	63.83
XLNet	110M (55X)	0.8582 (-0.28%)	74.25
BERT-Base	**110M (55X)**	**0.8803 (-2.78%)**	69.17
RoBERTa	125M (62X)	0.7644 (+11.96%)	58.17
ELECTRA	14M (7X)	0.8689 (-1.50%)	66.48
Distill BERT	66M (33X)	0.8707 (-1.71%)	67.18

6.3 Baselines

We fine tune our pre-trained model (CB3SA+PT) on the entire training dataset and compare it's performance against several strong baselines; XLNet, BERT, RoBERTa, DistillBERT, and ELECTRA - all fine tuned on the same training set. Table 4 lists the results. Additionally, the F1 column shows the percentage by which our F1 lags the respective models and the Params column shows the factor by which our parameter set is smaller.

Our model is able to use 55X lesser parameters than BERT and yet obtain a comparable performance, lagging BERT Base by 2.78% of F1 score on the test set. Among the SOTA models, ELECTRA-Small seems to provide the best parameter efficiency by uisng 7.8X lesser parameters than BERT-Base while lagging by only 1.3% on F1 score on the test set. Interestingly, RoBERTa, the biggest model that we tried with 125M parameters, under performed by a huge margin, lagging our F1 score by 11.96%.

6.4 Low Resource Setting

We experiment in a low resource setting by allowing all the models to be fine tuned only on a small fraction of the training data and then evaluating their performance on the same test dataset as was used in Section 6.3. Figure 2 compares the performance of our pre trained model with that of the SOTA models which shows our model's performance deteriorates much lesser compared to others. **When allowed to be fine tuned only on 5% of the training data (around 1600 parallel examples) our model outperforms the best performing SOTA model, BERT-Base, by 24.3%.** In fact, our model continues outperforming all SOTA models for a large part of the low data regime, till 20%, after which others catch up.

7 CONCLUSION

We propose a self-attentive recurrent neural network architecture for product title compression. We successfully pre train our network

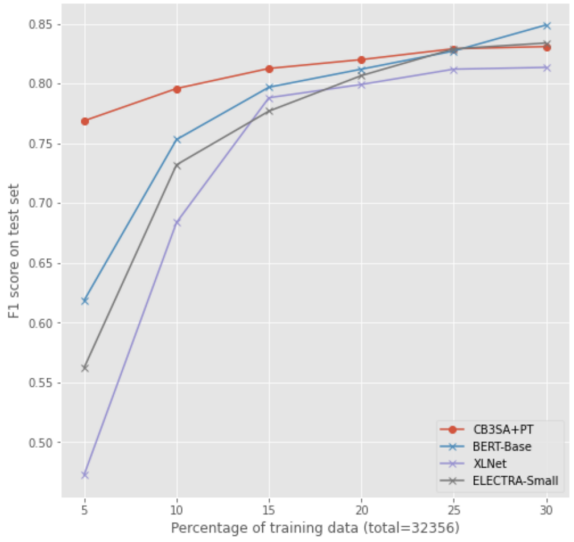

Figure 2: F1 score on test set when allowed to fine tune only on a small fraction of training data. Parameter heavy state-of-the-art models rapidly deteriorate as amount of training data is reduced while our pre trained model is resilient in the low data regime.

Table 5: Sample title compression by CB3SA+PT

Long Title	Short Title
Glad OdorShield Tall Kitchen Drawstring Trash bags - Febreze Fresh Clean - 13 Gallon - 40 count (Packaging May Vary)	trash bags
Great Value Strawberry Nonfat Greek Yogurt, 6 oz, 4 ct	nonfat greek yogurt
Del Monte Fresh Cut Cut Green Beans & Potatoes With Ham Style Flavor, 29 Oz	green beans and potatoes
Suave Professionals Moisturizing Shampoo and Conditioner Almond + Shea Butter 28 oz, 2 count	shampoo and conditioner

as a discriminator on a replaced token detection task on unlabeled dataset of long product titles. We also fine tune several large state of the art pre trained NLP models for the title summarization task. Our experiments with human generated short titles on the Walmart grocery catalog show that our pre trained model with 2 Million parameters achieves performance comparable to to state of the art models like BERT and XLNet that use in excess of 100 Million parameters. Further, in a low resource setting when very small amount of training data is available, our model outperforms all SOTA models by a large margin.

REFERENCES

[1] Mehdi Allahyari, Seyed Amin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. 2017. Text Summarization Techniques: A Brief Survey. *CoRR* abs/1707.02268 (2017). arXiv:1707.02268 http://arxiv.org/abs/1707.02268

[2] Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca J. Passonneau. 2015. Abstractive Multi-Document Summarization via Phrase Selection and Merging. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. 1587–1597. https://www.aclweb.org/anthology/P15-1153/

[3] Jianpeng Cheng and Mirella Lapata. 2016. Neural Summarization by Extracting Sentences and Words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. https://www.aclweb.org/anthology/P16-1046/

[4] Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*. 93–98. https://www.aclweb.org/anthology/N16-1012/

[5] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=r1xMH1BtvB

[6] Vipul Dalal and Latesh G. Malik. 2013. A Survey of Extractive and Abstractive Text Summarization Techniques. In *6th International Conference on Emerging Trends in Engineering and Technology, ICETET 2013, 16-18 December, 2013, Nagpur, India*. 109–110. https://doi.org/10.1109/ICETET.2013.31

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 4171–4186. https://doi.org/10.18653/v1/n19-1423

[8] Thibault Févry and Jason Phang. 2018. Unsupervised Sentence Compression using Denoising Auto-Encoders. In *Proceedings of the 22nd Conference on Computational Natural Language Learning, CoNLL 2018, Brussels, Belgium, October 31 - November 1, 2018*. 413–422. https://www.aclweb.org/anthology/K18-1040/

[9] Katja Filippova, Enrique Alfonseca, Carlos A. Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence Compression by Deletion with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*. 360–368. https://www.aclweb.org/anthology/D15-1042/

[10] Katja Filippova and Yasemin Altun. 2013. Overcoming the Lack of Parallel Data in Sentence Compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*. 1481–1491. https://www.aclweb.org/anthology/D13-1155/

[11] Mahak Gambhir and Vishal Gupta. 2017. Recent automatic text summarization techniques: a survey. *Artif. Intell. Rev.* 47, 1 (2017), 1–66. https://doi.org/10.1007/s10462-016-9475-9

[12] Yu Gong, Xusheng Luo, Kenny Q. Zhu, Wenwu Ou, Zhao Li, and Lu Duan. 2019. Automatic Generation of Chinese Short Product Titles for Mobile Display. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 9460–9465. https://doi.org/10.1609/aaai.v33i01.33019460

[13] Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, 328–339. https://doi.org/10.18653/v1/P18-1031

[14] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1746–1751. https://doi.org/10.3115/v1/D14-1181

[15] Yishu Miao and Phil Blunsom. 2016. Language as a Latent Variable: Discrete Generative Models for Sentence Compression. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. 319–328. https://www.aclweb.org/anthology/D16-1031/

[16] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*. 3075–3081. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14636

[17] Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*. 280–290. https://www.aclweb.org/anthology/K16-1028/

[18] Ani Nenkova and Kathleen R. McKeown. 2012. A Survey of Text Summarization Techniques. In *Mining Text Data*. 43–76. https://doi.org/10.1007/978-1-4614-3223-4_3

[19] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional Attention Flow for Machine Comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=HJ0UKP9ge

[20] Tian Shi, Yaser Keneshloo, Naren Ramakrishnan, and Chandan K. Reddy. 2018. Neural Abstractive Text Summarization with Sequence-to-Sequence Models. *CoRR* abs/1812.02303 (2018). arXiv:1812.02303 http://arxiv.org/abs/1812.02303

[21] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway Networks. *CoRR* abs/1505.00387 (2015). arXiv:1505.00387 http://arxiv.org/abs/1505.00387

[22] Fei Sun, Peng Jiang, Hanxiao Sun, Changhua Pei, Wenwu Ou, and Xiaobo Wang. 2018. Multi-Source Pointer Network for Product Title Summarization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, Alfredo Cuzzocrea, James Allan, Norman W. Paton, Divesh Srivastava, Rakesh Agrawal, Andrei Z. Broder, Mohammed J. Zaki, K. Selçuk Candan, Alexandros Labrinidis, Assaf Schuster, and Haixun Wang (Eds.). ACM, 7–16. https://doi.org/10.1145/3269206.3271722

[23] Lai Dac Viet, Nguyen Truong Son, and Le Minh Nguyen. 2017. Deletion-Based Sentence Compression Using Bi-enc-dec LSTM. In *Computational Linguistics - 15th International Conference of the Pacific Association for Computational Linguistics, PACLING 2017, Yangon, Myanmar, August 16-18, 2017, Revised Selected Papers*. 249–260. https://doi.org/10.1007/978-981-10-8438-6_20

[24] Jingang Wang, Junfeng Tian, Long Qiu, Sheng Li, Jun Lang, Luo Si, and Man Lan. 2018. A Multi-Task Learning Approach for Improving Product Title Compression with User Search Log Data. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 451–458. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16766

[25] Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, and Claire Cardie. 2013. A Sentence Compression Based Framework to Query-Focused Multi-Document Summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*. 1384–1394. https://www.aclweb.org/anthology/P13-1136/

[26] Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-Sequence Learning as Beam-Search Optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. 1296–1306. https://www.aclweb.org/anthology/D16-1137/

[27] Joan Xiao and Robert Munro. 2019. Text Summarization of Product Titles. In *Proceedings of the SIGIR 2019 Workshop on eCommerce, co-located with the 42st International ACM SIGIR Conference on Research and Development in Information Retrieval, eCom@SIGIR 2019, Paris, France, July 25, 2019 (CEUR Workshop Proceedings, Vol. 2410)*, Jon Degenhardt, Surya Kallumadi, Utkarsh Porwal, and Andrew Trotman (Eds.). CEUR-WS.org. http://ceur-ws.org/Vol-2410/paper36.pdf

[28] Jianguo Zhang, Pengcheng Zou, Zhao Li, Yao Wan, Ye Liu, Xiuming Pan, Yu Gong, and Philip S. Yu. 2018. Product Title Refinement via Multi-Modal Generative Adversarial Learning. *CoRR* abs/1811.04498 (2018). arXiv:1811.04498 http://arxiv.org/abs/1811.04498

[29] Yang Zhao, Zhiyuan Luo, and Akiko Aizawa. 2018. A Language Model based Evaluator for Sentence Compression. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*. 170–175. https://doi.org/10.18653/v1/P18-2028