Description Based Text Classification with Reinforcement Learning

Duo Chai [1] **Wei Wu** [1] **Qinghong Han** [1] **Wu Fei** [2] **Jiwei Li** [1]

Abstract

The task of text classification is usually divided into two stages: *text feature extraction* and *classification*. In this standard formalization, categories are merely represented as indexes in the label vocabulary, and the model lacks for explicit instructions on what to classify. Inspired by the current trend of formalizing NLP problems as question answering tasks, we propose a new framework for text classification, in which each category label is associated with a category description. Descriptions are generated by hand-crafted templates or using abstractive/extractive models from reinforcement learning. The concatenation of the description and the text is fed to the classifier to decide whether or not the current label should be assigned to the text. The proposed strategy forces the model to attend to the most salient texts with respect to the label description, which can be regarded as a hard version of attention, leading to better performances. We observe significant performance boosts over strong baselines on a wide range of text classification tasks including single-label classification, multi-label classification and multi-aspect sentiment analysis.

1. Introduction

Text classification (Kim, 2014; Joulin et al., 2016; Yang et al., 2016) is a fundamental problem in natural language processing. The task is to assign one or multiple category label(s) to a sequence of text tokens. It has broad applications such as sentiment analysis (Pang et al., 2002; Maas et al., 2011; Socher et al., 2013; Tang et al., 2014; 2015b), aspect sentiment classification (Jo & Oh, 2011; Tang et al., 2015a; Wang et al., 2015; Nguyen & Shirai, 2015; Tang et al., 2016b; Pontiki et al., 2016; Sun et al., 2019b), topic classification (Schwartz et al., 1997; Quercia et al., 2012; Wang & Manning, 2012), spam detection (Ott et al., 2011; 2013; Li et al., 2014), etc.

Standardly, text classification is divided into the following two steps: (1) *text feature extraction*: a sequence of texts is mapped to a feature representation based on handcrafted features such as bag of words (Pang et al., 2002), topics (Blei et al., 2003; Mcauliffe & Blei, 2008), or distributed vectors using neural models such as LSTMs (Hochreiter & Schmidhuber, 1997), CNNs (Kalchbrenner et al., 2014; Kim, 2014) or recursive nets (Socher et al., 2013; Irsoy & Cardie, 2014; Li et al., 2015; Bowman et al., 2016); and (2) *classification*: the extracted representation is fed to a classifier such as SVM, logistic regression or the softmax function to output the category label.

This standard formalization for the task of text classification has an intrinsic drawback: categories are merely represented as indexes in the label vocabulary, and lack for explicit instructions on what to classify. Labels can only influence the training process when the supervision signals are back propagated to feature vectors extracted from the feature extraction step. Class indicators in the text, which might just be one or two keywords, could be deeply buried in the huge chunk of text, making it hard for the model to separate grain from chaff. Additionally, signals for different classes might entangle in the text. Take the task of aspect sentiment classification (Lei et al., 2016) as an example, the goal of which is to classify the sentiment of a specific aspect of a review. A review might contain diverse sentiments towards different aspects and that they are entangled together, *e.g. "clean updated room. friendly efficient staff . rate was too high."*. Under the standard formalization, the label of a text sequence is merely an index indicating the sentiment of a predefined but not explicitly mentioned aspect from the view of the model. The model needs to first learn to associate the relevant text with the target aspect, and then decide the sentiment, which inevitably adds to the difficulty.

Inspired by the current trend of formalizing NLP problems as question answering tasks (Levy et al., 2017; McCann et al., 2018; Li et al., 2019a;b; Gardner et al., 2019; Raffel et al., 2019), we propose a new framework for text classification by formalizing it as a SQuAD-style machine reading comprehension task. The key point for this formalization is to associate each class with a class description to explicitly tell the model what to classify. For example, the task of classifying hotel reviews with positive location in aspect sentiment classification for review $\boldsymbol{x} = \{x_1, x_2, ..., x_n\}$ is

[1] Shannon.AI [2] Zhejiang University. Correspondence to: Jiwei Li <jiwei_li@shannonai.com>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

transformed to assigning a "yes/no" label to "[CLS] positive location [SEP] $\boldsymbol{x}$", indicating whether the attribute towards the location of the hotel in review $\boldsymbol{x}$ is positive. By explicitly mentioning what to classify, the incorporation of class description forces the model to attend to the most salient texts with respect to the label, which can be regarded as a hard version of attention. This strategy provides a straightforward resolution to the issues mentioned in the previous paragraph.

One key issue with this method is how to obtain category descriptions. Recent models that cast NLP problems as QA tasks (Li et al., 2019a;b; Gardner et al., 2019) use handcrafted templates to generate descriptions, and have two major drawbacks: (1) it is labor-intensive to predefine descriptions for each category, especially when the number of category is large; and (2) the model performance is sensitive to how the descriptions are constructed and human-generated templates might be sub-optimal. To handle this issue, we propose to automatically generate descriptions using reinforcement learning. The description can be generated in an extractive way, extracting a substring of the input text and using it as the description, or in an abstractive way, using generative model to generate a string of tokens and using it as the description. The model is trained in an end-to-end fashion to jointly learn to generate proper class descriptions and to assign correct class labels to texts.

We are able to observe significant performance boosts against strong baselines on a wide range of text classification benchmarks including single-label classification, multi-label classification and multi-aspect sentiment analysis.

2. Related Work

2.1. Text Classification

Neural models such as CNNs (Kim, 2014), LSTMs (Hochreiter & Schmidhuber, 1997; Tang et al., 2016a), recursive nets (Socher et al., 2013) or Transformers (Vaswani et al., 2017; Devlin et al., 2019), have been shown to be effective in text classification. Joulin et al. (2017); Bojanowski et al. (2017) proposed fastText, representing the whole text using the average of embeddings of constituent words.

There has been work investigating the rich information behind class labels. In the literature of zero-shot text classification, knowledge of labels are incorporated in the form of word embeddings (Yogatama et al., 2017; Rios & Kavuluru, 2018), or class descriptions (Zhang et al., 2019; Srivastava et al., 2018). Wang et al. (2018a) proposed a label-embedding attentive model that jointly embeds words and labels in the same latent space, and the text representations are constructed directly using the text-label compatibility. Sun et al. (2019a) constructed auxiliary sentences from the aspect in the task of aspect based sentiment analysis (ABSA) by using four different sentence templates, and thus converted ABSA to a sentence-pair classification task. Wang et al. (2019) proposed to frame ABSA towards question answering (QA), and designed an attention network to select aspect-specific words, which alleviates the effects of noisy words for a specific aspect. Descriptions in Sun et al. (2019a) and Wang et al. (2019) are generated from crowd-sourcing. This work takes a major step forward, in which the model is able to learn to automatically generate proper label descriptions from reinforcement learning.

2.2. Formalizing NLP Tasks as Question Answering

Question Answering MRC models (Rajpurkar et al., 2016; Seo et al., 2016; Wang et al., 2016; Wang & Jiang, 2016; Xiong et al., 2016; 2017; Wang et al., 2016; Shen et al., 2017; Chen et al., 2017b; Rajpurkar et al., 2018) extract answer spans from passages given questions. The task can be formalized as two multi-class classification tasks, i.e., predicting the starting and ending positions of the answer spans given questions. The context can either be prepared in advance (Seo et al., 2017) or selected from a large scale open-domain corpus such as Wikipedia (Chen et al., 2017a).

Query Generation In the standard version of MRC QA systems, queries are defined in advance. Some of recent works have studied how to generate queries for better answer extraction. Yuan et al. (2017) combines supervised learning and reinforcement learning to generate natural language descriptions; Yang et al. (2017) trained a generative model to generate queries based on unlabeled texts to train QA models; Du et al. (2017) framed the task of description generation as a *seq2seq* task, where descriptions are generated conditioning on the texts; Zhao et al. (2018) utilized the copy mechanism (Gu et al., 2016; Vinyals et al., 2015) and Kumar et al. (2018) proposed a generator-evaluator framework that directly optimizes objectives. Our work is similar to Yuan et al. (2017) and Kumar et al. (2018) in terms of description generation, in which reinforcement learning is applied for description/query generation.

Formalizing NLP tasks as QA There has recently been a trend of casting NLP problem as QA tasks. Gardner et al. (2019) posed three motivations for using question answering as a format for a particular task, *i.e.*, to fill human information needs, to probe a system's understanding of some context and to transfer learned parameters from one task to another. , Levy et al. (2017) transformed the task of relation extraction to a QA task, in which each relation type $r(\boldsymbol{x}, \boldsymbol{y})$ is characterized as a question $q(\boldsymbol{x})$ whose answer is $\boldsymbol{y}$. In a followup, Li et al. (2019b) formalized the task of entity-relation extraction as a multi-turn QA task by utilizing a template-based procedure to construct descriptions for relations and extract pairs of entities between which a relation holds. Li et al. (2019a) introduced a QA framework for the task of named entity recognition, in which the extraction of an entity within the text is formalized as answering questions like ”*which person is mentioned in the text?*”. McCann et al. (2018) built a multi-task question answering network for different NLP tasks, for example, the generation of a summary given a chunk of text is formalized as answering the question “*What is the summary?*”. Wu et al. (2019) formalized the task of coreference as a question answering task.

3. description-based Text Classification

Consider a sequence of text $x = \{x_1, \cdots, x_L\}$ to classify, where L denotes the length of the text $\boldsymbol{x}$. Each x is associated with a class label $y \in \mathcal{Y} = [1, N]$, where N denotes the number of the predefined classes. It is worth noting that in the task of single-label classification, y can take only one value. While for the multi-label classification task, y can take multiple values.

We use BERT (Devlin et al., 2019) as the backbone to illustrate how the proposed method works. It is worth noting that the proposed method is a general one and can be easily extended to other model bases with minor adjustments. Under the formalization of the description-based text classification, each class y is associated with a unique natural language description $\boldsymbol{q}_y = \{q_{y1}, \cdots, q_{yL}\}$. The description encodes prior information about the label and facilitates the process of classification.

For an N-class multi-class classification task, empirically, one can train N binary classifiers or an N-class classifier, as will be separately described below.

N binary classifiers For the strategy of training N binary classifers, we iterate over all q_y to decide whether the label y should be assigned to a given instance x. More concretely, we first concatenate the text $\boldsymbol{x}$ and with the description $\boldsymbol{q}_y$ to generate $\{[\text{CLS}]; \boldsymbol{q}_y; [\text{SEP}]; \boldsymbol{x}\}$, where [CLS] and [SEP] are special tokens. Next, the concatenated sequence is fed to transformers in BERT, from which we we obtain the contextual representations $h_{[\text{CLS}]}$. Now that the representation $h_{[\text{CLS}]}$ has encoded interactions between the text and the description, another two-layer feed forward network is used to transform $h_{[\text{CLS}]}$ to a real value between 0 and 1 by using the sigmoid function, representing the probability of label y being assigned to the text $\boldsymbol{x}$, as follows:

$$p(y|\boldsymbol{x}) = \text{sigmoid}(W_2\text{ReLU}(W_1 h_{[\text{CLS}]} + b_1) + b_2) \quad (1)$$

where W_1, W_2, b_1, b_2 are parameters to optimize. At test time, for a multi-label classification task, in which multiple labels can be assigned to an instance, the resulting label set is as follows:

$$\tilde{\boldsymbol{y}} = \{y \mid p(y|\boldsymbol{x}) > 0.5, \forall y \in \mathcal{Y}\} \quad (2)$$

and for single-label classification, the resulting label set is as follows:

$$\tilde{\boldsymbol{y}} = \arg\max_y(\{p(y|\boldsymbol{x}), \forall y \in \mathcal{Y}\}) \quad (3)$$

One N-class classifier For the strategy of training an N-class classifier, we concatenate all descriptions with the input x, which is given as follows:

$$\{[\text{CLS1}]; \boldsymbol{q}_1; [\text{CLS2}]; \boldsymbol{q}_2; ...; [\text{CLS-N}]; \boldsymbol{q}_N; [\text{SEP}]; \boldsymbol{x}\}$$

where [CLSn] $1 \leq n \leq N$ are the special place-holding tokens. The concatenated input is then fed to the transformer, from which we obtain the the contextual representations $h_{[\text{CLS1}]}, h_{[\text{CLS2}]}, ..., h_{[\text{CLSN}]}$. The probability of assigning class n to instance x is obtained by first mapping $h_{[\text{CLSn}]}$ to scalars, and then outputting them to a softmax function, which is given as follows:

$$\begin{aligned} a_n &= \hat{h}^T \cdot h_{[\text{CLSn}]} \\ p(y = n|x) &= \frac{\exp{(a_n)}}{\sum_{t=1}^{t=N} \exp{(a_t)}} \end{aligned} \quad (4)$$

It is worth noting that the on N-class classifier strategy can not handle the multi-label classification case.

4. Description Construction

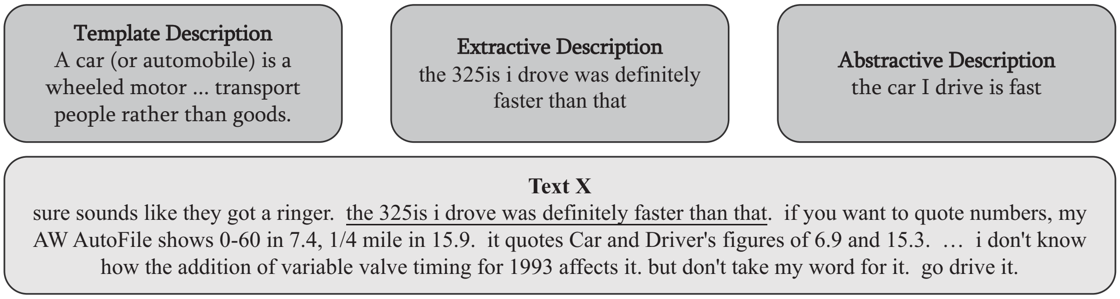
In this section, we describe the three proposed strategies to construct descriptions: the **template (Tem)** strategy (Section 4.1), the **extractive (Ext)** strategy (Section 4.2) and the **abstractive (Abs)** strategy (Section 4.3). An example of descriptions constructed by different strategies is shown in Figure 1.

4.1. The Template Strategy

As previous works (Li et al., 2019b;a; Levy et al., 2017) did, the most straightforward way to construct label descriptions is to use handcrafted templates. Templates can come from various sources, such as Wikipedia definitions, or human annotators. Example explnanations for some of the 20 news categoroes are shown in Table 1. More comprehensive template descriptions are listed in the supplementary material.

4.2. The Extractive Strategy

Generating descriptions using templates is suboptimal since (1) it is labor-intensive to ask humans to write down templates for different classes, especially when the number of classes is large; and (2) inappropriately constructed templates will actually lead to inferior performances, as demonstrated in Li et al. (2019a). The model should have the ability to learn to *generate* the most appropriate descriptions regarding different classes conditioning on the current text



Template Description
A car (or automobile) is a wheeled motor ... transport people rather than goods.

Extractive Description
the 325is i drove was definitely faster than that

Abstractive Description
the car I drive is fast

Text X
sure sounds like they got a ringer. the 325is i drove was definitely faster than that. if you want to quote numbers, my AW AutoFile shows 0-60 in 7.4, 1/4 mile in 15.9. it quotes Car and Driver's figures of 6.9 and 15.3. … i don't know how the addition of variable valve timing for 1993 affects it. but don't take my word for it. go drive it.

Figure 1. An example of descriptions constructed via different strategies. Text is from the 20news dataset.

Label	Description
COMP.SYS.MAC.HARDWARE	The Macintosh is a family of personal computers designed ... since January 1984.
REC.AUTOS	A car (or automobile) is a wheeled motor ... transport people rather than goods.
TALK.POLITICS.MISC	Politics is a set of activities ... making decisions that apply to groups of members.

Table 1. Examples of template descriptions drawn from Wikipedia for the `20news` dataset. For other labels and datasets, we also use their Wikipedia definitions as template descriptions.

to classify, and the appropriateness of the generated descriptions should directly correlate with the final classification performance. To this end, we describe two ways to generate descriptions, the *extractive* strategy, as will be detailed in this subsection, and the *abstractive* strategy, which will be detailed in the next subsection.

For the extractive strategy, for each input $\boldsymbol{x} = \{x_1, \cdots, x_T\}$, the extractive model generates a description $\boldsymbol{q}_{yx}$ for each class label y, where $\boldsymbol{q}_{yx}$ is a substring of $\boldsymbol{x}$. As can be seen, for different inputs x, the descriptions for the same class can be different. For the golden class y that should be assigned to x, there should be a substring of x relevant to y, and this substring will be chosen as the description for y. But classes that should not be assigned, there might not be corresponding substrings in x that can be used as descriptions. To deal with this issue, we append N dummy tokens to x, providing the model the flexibility of handling the case where there is no corresponding substring within x to a class label. If the extractive model picks a dummy token, it will use hand-crafted templates for different categories as descriptions.

To back-propagate the signal indicating which span contributes how much to the classification performance, we turn to reinforcement learning, an approach that encourages the model to act toward higher rewards. A typical reinforcement learning algorithm consists of three components: the action a, the policy π and the reward R.

Action and Policy For each class label y, the action is to pick a text span $\{x_{i_s}, \cdots, x_{i_e}\}$ from $\boldsymbol{x}$ to represent $\boldsymbol{q}_{yx}$. Since a span is a sequence of continuous tokens in the text, we only need to select the starting index i_s and the ending index i_e, denoted by a_{i_s,i_e}.

For each class label y, the policy π defines the probability of selecting the starting index i_s and the ending index i_e. Following previous work (Chen et al., 2017a; Devlin et al., 2019), each token x_k within $\boldsymbol{x}$ is mapped to a representation h_k using BERT, and the probability of x_i being the starting index and the ending index of $\boldsymbol{q}_{yx}$ are given as follows:

$$\begin{aligned} P_{\text{start}}(y,k) &= \frac{\exp(W^{ys} h_k)}{\sum_{t=1}^{t=T} \exp(W^{ys} h_t)} \\ P_{\text{end}}(y,k) &= \frac{\exp(W^{ye} h_k)}{\sum_{t=1}^{t=T} \exp(W^{ye} h_t)} \end{aligned} \tag{5}$$

where W^{ys} and W^{ye} are $1 \times K$ dimensional vectors to map h_t to a scalar. Each class y has a class-specific W^{ys} and W^{ye}. The probability of a text span with the starting index i_s and ending index i_e being the description for class y , denoted by $P_{\text{span}}(y, a_{i_s,i_e})$, is given as follows:

$$P_{\text{span}}(y, a_{i_s,i_e}) = P_{\text{start}}(y, i_s) \times P_{\text{end}}(y, i_e) \tag{6}$$

Reward Given $\boldsymbol{x}$ and a description $\boldsymbol{q}_{yx}$, the classification model in Section 3 will output the probability of assigning the correct label to x, which will be used as the reward to update both the classification model and the extractive model. Specifically, for multi-class classification, all $\boldsymbol{q}_{yx}$ are concatenated with $\boldsymbol{x}$, and the reward is given as follows

$$R(x, \boldsymbol{q}_{yx} \text{for all y}) = p(y = n|x) \tag{7}$$

where n is the gold label for x.

For N-binary-classification model, each $\boldsymbol{q}_{yx}$ is separately concatenated with x, and the reward is given as follows:

$$R(x, \boldsymbol{q}_{yx}) = p(y = \hat{y}|x) \tag{8}$$

where $\hat{y}$ is the golden binary label. [1]

REINFORCE To find the optimal policy, we use the REINFORCE algorithm (Williams, 1992), a kind of policy gradient method which maximizes the expected reward $\mathbb{E}_\pi[R(x, \boldsymbol{q}_y)]$. For each generated description $\boldsymbol{q}_{yx}$ and the corresponding x, we define its loss as follows:

$$\mathcal{L} = -\mathbb{E}_\pi[R(\boldsymbol{q}_{yx}, x)] \tag{9}$$

REINFORCE approximates the expectation in Eq. 9 with sampled descriptions from the policy distribution. The gradient to update parameters is given as follows:

$$\nabla\mathcal{L} \approx -\sum_{i=1}^{B} \nabla \log \pi(a_{i_s,i_e}|\boldsymbol{x}, y)[R(\boldsymbol{q}_y) - b] \tag{10}$$

where b denotes the baseline value, which is set to the average of all previous rewards. The extractive policy is initialized to generate dummy tokens as descriptions. Then the extractive model and the classification model are jointly trained based on the reward.

4.3. The Abstractive Strategy

An alternative generation strategy is to generate descriptions using generation models. The generation model uses the sequence-to-sequence structure (Sutskever et al., 2014; Vaswani et al., 2017) as a backbone. It takes x as an input, and generate different descriptions q_{yx} for different x.

Action and Policy For each class label y, the action is to generate the description $\boldsymbol{q}_{yx} = \{q_1, \cdots, q_L\}$, defined by p_θ. The policy P_{SEQ2SEQ} defines the probability of generating the entire string of the description given x, which is equivalent to generating each token within the description, and is given as follows:

$$P_{\text{SEQ2SEQ}}(\boldsymbol{q}_y|x) = \prod_{i=1}^{L} p_\theta(q_i|q_{<i}, x, y) \tag{11}$$

where $q_{<i}$ denotes all the already generated tokens. $P_{\text{SEQ2SEQ}}(\boldsymbol{q}_y|x)$ for different class y share the structures and parameters, with the only difference being that a class-specific embedding h_y is appended to each source and target token.

Reward The RL reward and the training loss for the abstractive strategy are similar to those for the extractive strategy, as in Eq. 7 and in Eq. 9. A widely recognized challenge for training language models using RL is the high variance, since the action space is huge (Ranzato et al., 2015; Yu et al., 2017; Li et al., 2017). To deal with this issue, we use the REGS – Reward for Every Generation Step proposed by Li et al. (2017). Unlike standard REINFORCE training, in which the same reward is used to update the probability of all tokens within the description, REGS trains a a discriminator that is able to assign rewards to partially decoded sequences. The gradient is given by:

$$\nabla\mathcal{L} \approx -\sum_{i=1}^{L} \nabla \log \pi(q_i|q_{<i}, h_y)[R(q_{<i}) - b(q_{<i})] \tag{12}$$

Here $R(q_{<i})$ denotes the reward given the partially decoded sequence $q_{<i}$ as the description, and $b(q_{<i})$ denotes the baseline.

The generative policy P_{SEQ2SEQ} is initialized using a pre-trained encoder-decoder with input being x and output being template descriptions. Then the description generation model and the classification model are jointly trained based on the reward.

5. Experiments

5.1. Benchmarks

We use the following widely used benchmarks to test the proposed model. The detailed descriptions for benchmarks are found in the supplementary material.

- **Single-label Classification**: The task of single-label classification is to assign a single class label to the text to classify. We use the following widely used benchmarks: (1) **AGNews**: Topic classification over four categories of Internet news articles (Del Corso et al., 2005). (2) **20newsgroups**[2]: The 20 Newsgroups data set is a collection of documents over 20 different newsgroups. (3) **DBPedia**: Ontology classification over fourteen non-overlapping classes picked from DBpedia 2014 (Wikipedia). (4) **Yahoo! Answers**: Topic classification ten largest main categories from Yahoo! Answers. (5) **Yelp Review Polarity (YelpP)**: Binary sentiment classification over yelp reviews. (6) **IMDB**: Binary sentiment classification over IMDB reviews.
- **Multi-label Classification**: The goal of multi-label classification is to assign multiple class labels to a single text. We use (1) **Reuters**[3]: A multi-label benchmark dataset for document classification with 90 classes. (2) **AAPD**: The arXiv Academic Paper dataset (Yang et al., 2018) with 54 classes.

[1] Experiments show that using the probability as the reward performs better than using the log probability.

[2] http://qwone.com/~jason/20Newsgroups/

[3] https://martin-thoma.com/nlp-reuters/

Table 2. Test error rates on the AGNews, 20news, DBPedia, Yahoo, Yelp P and IMDB datasets for single-label classification.

Model	AGNews	20news	DBPedia	Yahoo	YelpP	IMDB
Char-level CNN (Zhang et al., 2015)	8.5	–	1.4	28.8	4.4	–
VDCNN (Conneau et al., 2016)	8.7	–	1.3	26.6	4.3	–
DPCNN (Johnson & Zhang, 2017)	6.9	–	0.91	23.9	2.6	–
Label Embedding (Wang et al., 2018b)	7.5	–	1.0	22.6	4.7	–
LSTMs (Zhang et al., 2015)	13.9	22.5	1.4	29.2	5.3	9.6
Hierarchical Attention (Yang et al., 2016)	11.8	19.6	1.6	24.2	5.0	8.0
D-LSTM(Yogatama et al., 2017)	7.9	–	1.3	26.3	7.4	–
Skim-LSTM (Seo et al., 2018)	6.4	–	–	–	–	8.8
BERT (Devlin et al., 2018)	5.9	16.9	0.72	22.7	2.4	6.8
Description (Tem.)	5.2	15.8	0.65	22.1	2.2	5.8
Description (Ext.)	**5.0**	15.6	**0.63**	22.0	2.1	**5.5**
Description (Abs.)	5.1	**15.4**	0.62	**21.8**	**2.0**	**5.5**

Table 3. Test error rates on the Reuters and AAPD datasets for multi-label classification.

Model	Reuters	AAPD
LSTMs (Zhang et al., 2015)	16.8	33.5
Hi-Attention (Yang et al., 2016)	13.9	30.3
Label-Emb (Wang et al., 2018b)	13.6	29.9
LSTM_{reg} (Adhikari et al., 2019a)	13.0	29.5
BERT (Adhikari et al., 2019b)	11.0	26.6
Description (Tem.)	10.3	25.9
Description (Ext.)	10.1	26.0
Description (Abs.)	**10.0**	**25.7**

Table 4. Test error rates on the BeerAdvocate (Beer), TripAdvisor (Trip) for multi-aspect sentiment classification.

Model	Beer	Trip
LSTMs (Zhang et al., 2015)	34.9	47.6
Hi-Attention (Yang et al., 2016)	33.3	42.2
Label-Emb (Wang et al., 2018b)	32.0	43.5
BERT (Devlin et al., 2018)	27.8	35.6
Description (Tem.)	17.4	18.1
Description (Ext.)	16.0	**17.0**
Description (Abs.)	**15.6**	17.6

- **Multi-aspect Sentiment Analysis**: The goal of the task is to test a model's ability to identify entangled sentiments for different aspects of a review. Each review might contain diverse sentiments towards different aspects. Widely used datasets include (1) the **BeerAdvocate** review dataset over aspects `appearance`, `smell`. Lei et al. (2016) processed the dataset by picking examples with less correlated aspects, leading to a de-correlated subset for each aspect (aroma) and `palate`. (2) the hotel **TripAdvisor** review (Li et al., 2016) over four aspects, *i.e.*, `service`, `cleanliness`, `location` and `rooms`. Li et al. (2016) processed the dataset by picking examples with less correlated aspects. There are three classes, positive, negative and neutral for both datasets.

5.2. Baselines

We implement the following widely-used models as baselines. Hyper-parameters for baselines are tuned on the development sets to enforce apple-to-apple comparison. In addition, we also copy results of models from relevant papers.

- **LSTM**: The vanilla LSTM model (Zhang et al., 2015), which first maps the text sequence to a vector using LSTMs (Hochreiter & Schmidhuber, 1997). For single-label datasets, the obtained document embeddings are output to the softmax layer. For multi-label datasets, we follow Adhikari et al. (2019b), in which each label is associated with a binary sigmoid function, and then the document embedding is fed to output the class label.
- **Hierarchical Attention** (Yang et al., 2016): The hierarchical attention model which uses word-level attention to obtain sentence embeddings and uses sentence-level attention to obtain document embeddings. We follow the strategy adopted in the LSTM model to handle multi-label tasks.
- **Label Embedding** : Model proposed by Wang et al. (2018b) that jointly learns the label embeddings and document embeddings.
- **BERT**: We use the BERT-base model (Devlin et al., 2018; Adhikari et al., 2019b) as the baseline. We follow the standard classification setup in BERT, in which the embedding of [CLS] is fed to a softmax layer to output the probability of a class being assigned to an instance.

We follow the strategy adopted in the LSTM model to handle multi-label tasks.

5.3. Results and Discussion

Table 2 presents the results for single-label classification tasks. The three proposed strategies consistently outperform the BERT baseline. Specifically, the template-based strategy outperforms BERT, decreasing error rates by *i.e.,* -0.7 on AGNews, -1.1 on 20news, -0.07 on DBPedia, -0.6 on Yahoo, -0.2 on YelpP and -1.0 on IMDB. The extractive and abstractive strategies consistently outperform the template-based strategy, which is because of their ability to automatically learn the proper descriptions. The extractive strategy performs better than the abstractive strategy on the AGNews and IMDB, but worse on the others.

Table 3 shows the results on the two multi-label classification datasets – Reuters and AAPD. Again, we observe performance gains over the BERT baseline on both datasets in terms of F1 score.

Table 4 shows the experimental results on the two multi-aspect sentiment analysis datasets BeerAdvocate and TripAdvisor. Surprisingly huge gains are observed on both datasets. Specifically, for BeerAdvocate, our method (Abs.) decreases the error rate from 27.8 to 15.6, and for TripAdvisor, our method (Ext.) decreases the error rate from 35.6 to 17.0. The explanation for this huge boost is as follows: both datasets are deliberately constructed in a way that each review contains aspects with opposite sentiments entangling with each other. This makes it extremely hard for the model to learn to jointly identify the target aspect and the sentiment. The incorporation of description gives the model the ability to directly attend to the relevant text, which leads to significant performance boost.

6. Ablation Studies and Analysis

In this section, we perform comprehensive ablation studies for better understand the model's behaviors. More examples of human-crafted descriptions and descriptions learned from RL are shown in the supplementary material.

6.1. Impact of Human Generated Templates

How to construct queries has a significant influence on the final results. In this subsection, we use the Yahoo! Answer dataset for illustration. We use different ways to construct template descriptions and test their influences.

- **Label Index**: the description is the index of a class, *i.e.* "one", "two", "three".
- **Keyword**: the description is the keyword extension of each category.
- **Keyword Expansion**: we use Wordnet to retrieve the synonyms of keywords and the description is their concatenation.
- **Wikipedia**: definitions drawn from Wikipedia.

Table 5. Results on 20news using different templates as descriptions.

Model	Error Rate
BERT	16.9
Template Description (Label Index)	16.8 (-0.1)
Template Description (Keyword)	16.4 (-0.5)
Template Description (Key Expansion)	16.2 (-0.7)
Template Description (Wiki)	15.8 (-1.1)

Results are shown in Table 5. As can be seen, the performance is sensitive to the way that descriptions are constructed. The performance for label index is very close to that of the BERT baseline. This is because label indexes do not carry any semantic knowledge about classes. One can think of the representations for label indexes similar to the vectors for different classes in the softmax layer, making the two models theoretically the same. Wikipedia outperforms Keyword since descriptions from Wikipedia carry more comprehensive semantic information for each class.

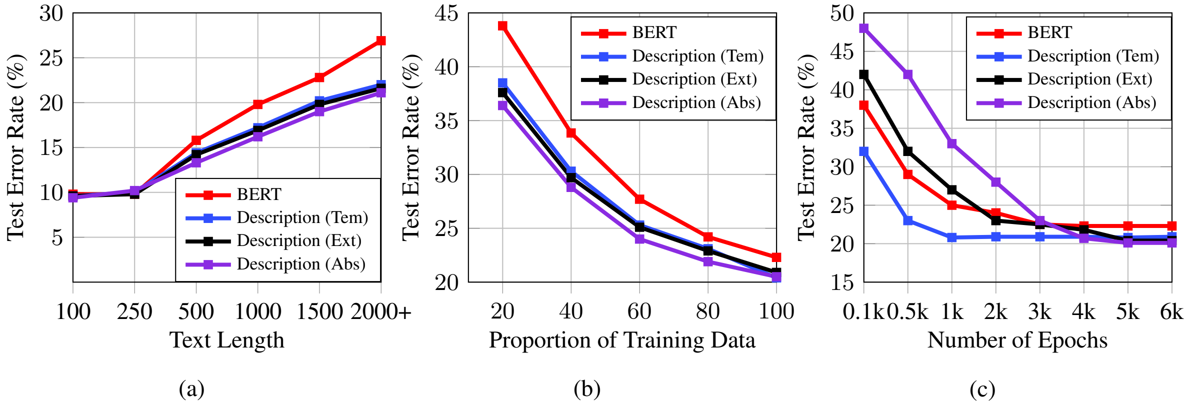
6.2. Impact on Examples with Different Lengths

It is interesting to see how differently the description-based models affect examples with different lengths. We use the IMDB dataset for illustrations since the IMDB dataset contains texts with more varying lengths. Since the model trained on the full set already has super low error rate (around 4-5%), we worry about the noise in comparison. We thus train different models on 20% of the training set, and test them on the test sets split into different buckets by text length.

Results are shown in Figure 2a. As can be seen, the superiority of description-based models over vanilla ones is more obvious on long texts. This is in line with our expectation: we can treat the descriptions as a hard version of attentions, forcing the model to look at the most relevant parts. For longer texts, where grain is mixed with larger amount of chaff, this mechanism will immediately introduce performance boosts. But for short texts, which is relatively easy for classification, both models can easily detect the relevant part and correctly classify it, making the gap smaller.

6.3. Convergence Speed

Figure 2c shows the convergence speed of different models on the Yahoo Answer dataset. For the description-based methods, the template model converges faster than the BERT baseline. This is because templates encode prior knowledge about categories. Instead of having the model learn to attend to the relevant texts, template-based methods force the model to pay attention to the relevant part. Both the abstractive strategy and the extractive strategy converge



(a)

(b)

(c)

Figure 2. (a) Test error rate vs text length (b) Test error rate vs proportion of training data (c) Test error rate vs the number of epochs.

Table 6. Error rates for different RL initialization strategies.

	Yahoo Answer	AAPD
Template	22.4	25.9
Ext (dummy Init)	22.2	26.0
Ext (ROUGE-L Init)	25.3	27.2
Ext (random Init)	28.0	30.1
Abs (template Init)	22.0	25.7
Abs (random Init)	87.9	78.4

slower than the template-based method and the BERT baseline. This is because it has to learn to generate the relevant description using reinforcement learning. Since the REINFORCE method is known for large variance, the model is slow to converge. The extractive strategy converges faster than the abstractive strategy due to the smaller search space.

6.4. Impact of the Size of Training Data

Since the description encodes prior semantic knowledge about categories, we expect that description-based methods work better with less training data. We trained different models on different proportions of the Yahoo Answer dataset, and test them on the original test set. From Figure 2b, we can see that the gap between the BERT baseline and the description-based models is significantly larger with 20% of training data and the gap is gradually narrowed down with increasing amount of training data.

6.5. Impact of RL Initialization Strategies

We explore the effect of different initialization strategies for RL on the Yahoo! Answer and AAPD datasets. For the extractive strategy, we explore random initialization and the ROUGE-L strategy. For the ROUGE-L strategy, the description for the correct label is the span that achieves the highest ROUGE-L score with respect to the template. The ROUGE-L strategy is widely used for sudo-golden answer/summary extraction in training extractive models, when golden answers are not substrings of the text in question answering (Nguyen et al., 2016) or golden summaries are not substrings of the input document (Kočiskỳ et al., 2018) for the task of summarization. The descriptions for incorrect labels are dummy tokens for the ROUGE-L strategy.

Results are shown in Table 6. As can be seen, generally, initialization matters. The extractive model is more immune to initialization strategies, even random initialization achieves acceptable performances. This is because of the smaller search space for extractive models relative to abstractive models. For the random initialization of the abstractive model, we are not able to make it converge within a reasonable amount of time.

7. Conclusion

We present a description-based text classification method that generates class-specific descriptions to give the model an explicit guidance of what to classify, which mitigates the issue of "meaningless labels". We develop three strategies to construct descriptions, *i.e.,* the template-based strategy, the extractive strategy and the abstractive strategy. The proposed framework achieves significant performance boost on a wide range of classification benchmarks.

References

Adhikari, A., Ram, A., Tang, R., and Lin, J. Rethinking complex neural network architectures for document classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4046–4051, Minneapolis, Minnesota, June 2019a. Association for Computational Linguistics. doi: 10.18653/v1/N19-1408.

Adhikari, A., Ram, A., Tang, R., and Lin, J. Docbert: Bert for document classification. *arXiv preprint arXiv:1904.08398*, 2019b.

Blei, D. M., Ng, A. Y., and Jordan, M. I. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan): 993–1022, 2003.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5 (0), 2017.

Bowman, S. R., Gauthier, J., Rastogi, A., Gupta, R., Manning, C. D., and Potts, C. A fast unified model for parsing and sentence understanding. *arXiv preprint arXiv:1603.06021*, 2016.

Chen, D., Fisch, A., Weston, J., and Bordes, A. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1870–1879, Vancouver, Canada, July 2017a. Association for Computational Linguistics. doi: 10.18653/v1/P17-1171.

Chen, D., Fisch, A., Weston, J., and Bordes, A. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*, 2017b.

Conneau, A., Schwenk, H., Barrault, L., and Lecun, Y. Very deep convolutional networks for natural language processing. *arXiv preprint arXiv:1606.01781*, 2, 2016.

Del Corso, G. M., Gulli, A., and Romani, F. Ranking a stream of news. In *Proceedings of the 14th international conference on World Wide Web*, pp. 97–106. ACM, 2005.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423.

Du, X., Shao, J., and Cardie, C. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1342–1352, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1123.

Gardner, M., Berant, J., Hajishirzi, H., Talmor, A., and Min, S. Question answering is a format; when is it useful?, 2019.

Gu, J., Lu, Z., Li, H., and Li, V. O. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1631–1640, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1154.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Irsoy, O. and Cardie, C. Deep recursive neural networks for compositionality in language. In *Advances in neural information processing systems*, pp. 2096–2104, 2014.

Jo, Y. and Oh, A. H. Aspect and sentiment unification model for online review analysis. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pp. 815–824. ACM, 2011.

Johnson, R. and Zhang, T. Deep pyramid convolutional neural networks for text categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 562–570, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1052.

Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 427–431, Valencia, Spain, April 2017. Association for Computational Linguistics.

Kalchbrenner, N., Grefenstette, E., and Blunsom, P. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Kim, Y. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181.

Kočiskỳ, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K. M., Melis, G., and Grefenstette, E. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018.

Kumar, V., Ramakrishnan, G., and Li, Y.-F. Putting the horse before the cart:a generator-evaluator framework for question generation from text, 2018.

Lei, T., Barzilay, R., and Jaakkola, T. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*, 2016.

Levy, O., Seo, M., Choi, E., and Zettlemoyer, L. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 333–342, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-1034.

Li, J., Ott, M., Cardie, C., and Hovy, E. Towards a general rule for identifying deceptive opinion spam. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1566–1576, 2014.

Li, J., Luong, M.-T., Jurafsky, D., and Hovy, E. When are tree structures necessary for deep learning of representations? *arXiv preprint arXiv:1503.00185*, 2015.

Li, J., Monroe, W., and Jurafsky, D. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016.

Li, J., Monroe, W., Shi, T., Jean, S., Ritter, A., and Jurafsky, D. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2157–2169, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1230.

Li, X., Feng, J., Meng, Y., Han, Q., Wu, F., and Li, J. A unified mrc framework for named entity recognition, 2019a.

Li, X., Yin, F., Sun, Z., Li, X., Yuan, A., Chai, D., Zhou, M., and Li, J. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1340–1350, Florence, Italy, July 2019b. Association for Computational Linguistics. doi: 10.18653/v1/P19-1129.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pp. 142–150. Association for Computational Linguistics, 2011.

Mcauliffe, J. D. and Blei, D. M. Supervised topic models. In *Advances in neural information processing systems*, pp. 121–128, 2008.

McCann, B., Keskar, N. S., Xiong, C., and Socher, R. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*, 2018.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. Ms marco: a human-generated machine reading comprehension dataset. 2016.

Nguyen, T. H. and Shirai, K. Phrasernn: Phrase recursive neural network for aspect-based sentiment analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2509–2514, 2015.

Ott, M., Choi, Y., Cardie, C., and Hancock, J. T. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pp. 309–319. Association for Computational Linguistics, 2011.

Ott, M., Cardie, C., and Hancock, J. T. Negative deceptive opinion spam. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: human language technologies*, pp. 497–501, 2013.

Pang, B., Lee, L., and Vaithyanathan, S. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pp. 79–86. Association for Computational Linguistics, 2002.

Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S., Mohammad, A.-S., Al-Ayyoub, M., Zhao, Y., Qin, B., De Clercq, O., et al. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pp. 19–30, 2016.

Quercia, D., Askham, H., and Crowcroft, J. Tweetlda: supervised topic classification and link prediction in twitter. In *Proceedings of the 4th Annual ACM Web Science Conference*, pp. 247–250. ACM, 2012.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264.

Rajpurkar, P., Jia, R., and Liang, P. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124.

Ranzato, M., Chopra, S., Auli, M., and Zaremba, W. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

Rios, A. and Kavuluru, R. Few-shot and zero-shot multi-label learning for structured label spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3132–3142, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1352.

Schwartz, R., Imai, T., Kubala, F., Nguyen, L., and Makhoul, J. A maximum likelihood model for topic classification of broadcast news. In *Fifth European Conference on Speech Communication and Technology*, 1997.

Seo, M., Kembhavi, A., Farhadi, A., and Hajishirzi, H. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*, 2016.

Seo, M., Min, S., Farhadi, A., and Hajishirzi, H. Neural speed reading via skim-RNN. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Sy-dQG-Rb.

Seo, M. J., Kembhavi, A., Farhadi, A., and Hajishirzi, H. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

Shen, Y., Huang, P.-S., Gao, J., and Chen, W. Reasonet: Learning to stop reading in machine comprehension. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1047–1055. ACM, 2017.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Srivastava, S., Labutov, I., and Mitchell, T. Zero-shot learning of classifiers from natural language quantification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 306–316, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1029.

Sun, C., Huang, L., and Qiu, X. Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 380–385, Minneapolis, Minnesota, June 2019a. Association for Computational Linguistics. doi: 10.18653/v1/N19-1035.

Sun, C., Huang, L., and Qiu, X. Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence. *arXiv preprint arXiv:1903.09588*, 2019b.

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

Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., and Qin, B. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1555–1565, 2014.

Tang, D., Qin, B., Feng, X., and Liu, T. Effective lstms for target-dependent sentiment classification. *arXiv preprint arXiv:1512.01100*, 2015a.

Tang, D., Qin, B., and Liu, T. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 1422–1432, 2015b.

Tang, D., Qin, B., Feng, X., and Liu, T. Effective LSTMs for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 3298–3307, Osaka, Japan, December 2016a. The COLING 2016 Organizing Committee.

Tang, D., Qin, B., and Liu, T. Aspect level sentiment classification with deep memory network. *arXiv preprint arXiv:1605.08900*, 2016b.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. 2017.

Vinyals, O., Fortunato, M., and Jaitly, N. Pointer networks. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 28*, pp. 2692–2700. 2015.

Wang, G., Li, C., Wang, W., Zhang, Y., Shen, D., Zhang, X., Henao, R., and Carin, L. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2321–2331, Melbourne, Australia, July 2018a. Association for Computational Linguistics. doi: 10.18653/v1/P18-1216.

Wang, G., Li, C., Wang, W., Zhang, Y., Shen, D., Zhang, X., Henao, R., and Carin, L. Joint embedding of words and labels for text classification. *arXiv preprint arXiv:1805.04174*, 2018b.

Wang, J., Sun, C., Li, S., Liu, X., Si, L., Zhang, M., and Zhou, G. Aspect sentiment classification towards question-answering with reinforced bidirectional attention network. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3548–3557, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1345.

Wang, S. and Jiang, J. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*, 2016.

Wang, S. and Manning, C. D. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th annual meeting of the association for computational linguistics: Short papers-volume 2*, pp. 90–94. Association for Computational Linguistics, 2012.

Wang, X., Sudoh, K., and Nagata, M. Rating entities and aspects using a hierarchical model. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 39–51. Springer, 2015.

Wang, Z., Mi, H., Hamza, W., and Florian, R. Multi-perspective context matching for machine comprehension. *arXiv preprint arXiv:1612.04211*, 2016.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696.

Wu, W., Wang, F., Yuan, A., Wu, F., and Li, J. Coreference resolution as query-based span prediction. *ArXiv*, abs/1911.01746, 2019.

Xiong, C., Zhong, V., and Socher, R. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604*, 2016.

Xiong, C., Zhong, V., and Socher, R. Dcn+: Mixed objective and deep residual coattention for question answering. *arXiv preprint arXiv:1711.00106*, 2017.

Yang, P., Sun, X., Li, W., Ma, S., Wu, W., and Wang, H. Sgm: sequence generation model for multi-label classification. *arXiv preprint arXiv:1806.04822*, 2018.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489, 2016.

Yang, Z., Hu, J., Salakhutdinov, R., and Cohen, W. Semi-supervised QA with generative domain-adaptive nets. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1040–1050, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1096.

Yogatama, D., Dyer, C., Ling, W., and Blunsom, P. Generative and discriminative text classification with recurrent neural networks, 2017.

Yu, L., Zhang, W., Wang, J., and Yu, Y. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Yuan, X., Wang, T., Gulcehre, C., Sordoni, A., Bachman, P., Zhang, S., Subramanian, S., and Trischler, A. Machine comprehension by text-to-text neural question generation. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pp. 15–25, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-2603.

Zhang, J., Lertvittayakumjorn, P., and Guo, Y. Integrating semantic knowledge to tackle zero-shot text classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1031–1040, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1108.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pp. 649–657, 2015.

Zhao, Y., Ni, X., Ding, Y., and Ke, Q. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3901–3910, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1424.