Rethinking Attribute Representation and Injection for Sentiment Classification

Abstract

Text attributes, such as user and product information in product reviews, have been used to improve the performance of sentiment classification models. The de facto standard method is to incorporate them as additional biases in the attention mechanism, and more performance gains are achieved by extending the model architecture. In this paper, we show that the above method is the least effective way to represent and inject attributes. To demonstrate this hypothesis, unlike previous models with complicated architectures, we limit our base model to a simple BiLSTM with attention classifier, and instead focus on how and where the attributes should be incorporated in the model. We propose to represent attributes as chunk-wise importance weight matrices and consider four locations in the model (i.e., embedding, encoding, attention, classifier) to inject attributes. Experiments show that our proposed method achieves significant improvements over the standard approach and that attention mechanism is the worst location to inject attributes, contradicting prior work. We also outperform the state-of-the-art despite our use of a simple base model. Finally, we show that these representations transfer well to other tasks. Model implementation and datasets are released here: this https URL.

Introduction

The use of categorical attributes (e.g., user, topic, aspects) in the sentiment analysis community BIBREF0, BIBREF1, BIBREF2 is widespread. Prior to the deep learning era, these information were used as effective categorical features BIBREF3, BIBREF4, BIBREF5, BIBREF6 for the machine learning model. Recent work has used them to improve the overall performance BIBREF7, BIBREF8, interpretability BIBREF9, BIBREF10, and personalization BIBREF11 of neural network models in different tasks such as sentiment classification BIBREF12, review summarization BIBREF13, and text generation

BIBREF8.

In particular, user and product information have been widely incorporated in sentiment classification models, especially since they are important metadata attributes found in review websites. BIBREF12 first showed significant accuracy increase of neural models when these information are used. Currently, the accepted standard method is to use them as additional biases when computing the weights a in the attention mechanism, as introduced by BIBREF7 as:

where u and p are the user and product embeddings, and h is a word encoding from BiLSTM. Since then, most of the subsequent work attempted to improve the model by extending the model architecture to be able to utilize external features BIBREF14, handle cold-start entities BIBREF9, and represent user and product separately BIBREF15.

Intuitively, however, this method is not the ideal method to represent and inject attributes because of two reasons. First, representing attributes as additional biases cannot model the relationship between the text and attributes. Rather, it only adds a user- and product-specific biases that are independent from the text when calculating the attention weights. Second, injecting the attributes in the attention mechanism means that user and product information are only used to customize how the model choose which words to focus on, as also shown empirically in previous work BIBREF7, BIBREF15. However, we argue that there are more intuitive locations to inject the attributes such as when contextualizing words to modify their sentiment intensity.

We propose to represent user and product information as weight matrices (i.e., W in the equation above). Directly incorporating these attributes into W leads to large increase in parameters and subsequently makes the model difficult to optimize. To mitigate these problems, we introduce chunk-wise importance weight matrices, which (1) uses a weight matrix smaller than W by a chunk size factor, and

(2) transforms these matrix into gates such that it corresponds to the relative importance of each neuron in W. We investigate the use of this method when injected to several locations in the base model: word embeddings, BiLSTM encoder, attention mechanism, and logistic classifier.

The results of our experiments can be summarized in three statements. First, our preliminary experiments show that doing bias-based attribute representation and attention-based injection is not an effective method to incorporate user and product information in sentiment classification models. Second, despite using only a simple BiLSTM with attention classifier, we significantly outperform previous state-of-the-art models that use more complicated architectures (e.g., models that use hierarchical models, external memory networks, etc.). Finally, we show that these attribute representations transfer well to other tasks such as product category classification and review headline generation.

How and Where to Inject Attributes?

In this section, we explore different ways on how to represent attributes and where in the model can we inject them.

How and Where to Inject Attributes? ::: The Base Model

The majority of this paper uses a base model that accepts a review $\mathbf {x}=x_1,...,x_n$ as input and returns a sentiment y as output, which we extend to also accept the corresponding user u and product p attributes as additional inputs. Different from previous work where models use complex architectures such as hierarchical LSTMs BIBREF7, BIBREF14 and external memory networks BIBREF16, BIBREF17, we aim to achieve improvements by only modifying how we represent and inject attributes. Thus, we use a simple classifier as our base model, which consists of four parts explained briefly as follows.

First, we embed $\mathbf {x}$ using a word embedding matrix that returns word embeddings $x^{\prime }_1,...,x^{\prime }_n$. We subsequently apply a non-linear function to each word:

Second, we run a bidirectional LSTM BIBREF18 encoder to contextualize the words into $h_t=[\overrightarrow{h}_t;\overleftarrow{h}_t]$ based on their forward and backward neighbors. The forward and backward LSTM look similar, thus for brevity we only show the forward LSTM below:

Third, we pool the encodings h_t into one document encoding d using attention mechanism BIBREF19, where v is a latent representation of informativeness BIBREF20:

Finally, we classify the document using a logistic classifier to get a predicted $y^{\prime }$:

Training is done normally by minimizing the cross entropy loss.

How and Where to Inject Attributes? ::: How: Attribute Representation

Note that at each part of the model, we see similar non-linear functions, all using the same form, i.e. $g(f(x)) = g(Wx + b)$, where $f(x)$ is an affine transformation function of x, g is a non-linear activation, W and b are weight matrix and bias parameters, respectively. Without extending the base model architecture, we can represent the attributes either as the weight matrix W or as the bias b to one of these functions by modifying them to accept u and p as inputs, i.e. $f(x,u,p)$.

How and Where to Inject Attributes? ::: How: Attribute Representation ::: Bias-based

The current accepted standard approach to represent the attributes is through the bias parameter b. Most of the previous work BIBREF7, BIBREF14, BIBREF9, BIBREF21 use Equation DISPLAY_FORM2 in the attention mechanism, which basically updates the original bias b to $b^{\prime } = W_u u + W_p p + b$. However, we argue that this is not the ideal way to incorporate attributes since it means we only add a user- and product-specific bias towards the goal of the function, without looking at the text. Figure FIGREF9 shows an intuitive example: When we represent user u as a bias in the logistic classifier, in which it means that u has a biased logits vector b_u of classifying the text as a certain sentiment (e.g., u tends to classify texts as three-star positive), shifting the final probability distribution regardless of what the text content may have been.

How and Where to Inject Attributes? ::: How: Attribute Representation ::: Matrix-based

A more intuitive way of representing attributes is through the weight matrix W. Specifically, given the attribute embeddings u and p, we linearly transform their concatenation into a vector $w^{\prime }$ of size D_1*D_2 where D_1 and D_2 are the dimensions of W. We then reshape $w^{\prime }$ into $W^{\prime }$ to get the same shape as W and replace W with $W^{\prime }$:

Theoretically, this should perform better than bias-based representations since direct relationship between text and attributes are modeled. For example, following the example above, $W^{\prime }x$ is a user-biased logits vector based on the document encoding d (e.g., u tends to classify texts as two-star positive when the text mentions that the dessert was sweet).

However, the model is burdened by a large number of parameters; matrix-based attribute representation increases the number of parameters by $|U|*|P|*D_1*D_2$, where $|U|$ and $|P|$ correspond to the number of users and products, respectively. This subsequently makes the weights difficult to optimize during training. Thus, directly incorporating attributes into the weight matrix may cause harm in the performance of the model.

How and Where to Inject Attributes? ::: How: Attribute Representation ::: CHIM-based

We introduce Chunk-wise Importance Matrix (CHIM) based representation, which improves over the matrix-based approach by mitigating the optimization problems mentioned above, using the following two tricks. First, instead of using a big weight matrix $W^{\prime }$ of shape (D_1, D_2), we use a chunked weight matrix C of shape $(D_1/C_1, D_2/C_2)$ where C_1 and C_2 are chunk size factors. Second, we use the chunked weight matrix as importance gates that shrinks the weights close to zero when they are deemed unimportant. We show the CHIM-based representation method in Figure FIGREF16.

We start by linearly transforming the concatenated attributes into c. Then we reshape c into C with shape $(D_1/C_1, D_2/C_2)$. These operations are similar to Equations DISPLAY_FORM14 and . We then repeat this matrix C_1*C_2 times and concatenate them such that we create a matrix $W^{\prime }$ of shape (D_1, D_2). Finally, we use the sigmoid function $\sigma $ to transform the matrix into gates that represent importance:

Finally we broadcast-multiply $W^{\prime }$ with the original weight matrix W to shrink the weights. The result is a sparse version of W, which can be seen as either a regularization step BIBREF22 where most weights are set close to zero, or a correction step BIBREF23 where the important gates are used to correct the weights. The use of multiple chunks regards CHIM as coarse-grained access control BIBREF24 where the use of different important gates for every node is unnecessary and expensive. The final function is shown below:

To summarize, chunking helps reduce the number of parameters while retaining the model performance, and importance matrix makes optimization easier during training, resulting to a performance improvement. We also tried alternative methods for importance matrix such as residual addition (i.e.,

$\tanh (W^{\prime }) + W$) introduced in BIBREF25, and low-rank adaptation methods BIBREF26, BIBREF27, but these did not improve the model performance.

How and Where to Inject Attributes? ::: Where: Attribute Injection

Using the approaches described above, we can inject attribute representation into four different parts of the model. This section describes what it means to inject attributes to a certain location and why previous work have been injecting them in the worst location (i.e., in the attention mechanism).

How and Where to Inject Attributes? ::: Where: Attribute Injection ::: In the attention mechanism

Injecting attributes to the attention mechanism means that we bias the selection of more informative words during pooling. For example, in Figure FIGREF9, a user may find delicious drinks to be the most important aspect in a restaurant. Injection in the attention mechanism would bias the selection of words such as wine, smooth, and sweet to create the document encoding. This is the standard location in the model to inject the attributes, and several BIBREF7, BIBREF9 have shown how the injected attention mechanism selects different words when the given user or product is different.

We argue, however, that attention mechanism is not the best location to inject the attributes. This is because we cannot obtain user- or product-biased sentiment information from the representation. In the example above, although we may be able to select, with user bias, the words wine and sweet in the text, we do not know whether the user has a positive or negative sentiment towards these words (e.g., Does the user like wine? How about sweet wines? etc.). In contrast, the three other locations we discuss below use the attributes to modify how the model looks at sentiment at different levels of textual granularity.

How and Where to Inject Attributes? ::: Where: Attribute Injection ::: In the word embedding

Injecting attributes to the word embedding means that we bias the sentiment intensity of a word independent from its neighboring context. For example, if a user normally uses the words tasty and delicious with a less and more positive intensity, respectively, the corresponding attribute-injected word embeddings would come out less similar, despite both words being synonymous.

How and Where to Inject Attributes? ::: Where: Attribute Injection ::: In the BiLSTM encoder

Injecting attributes to the encoder means that we bias the contextualization of words based on their neighbors in the text. For example, if a user likes their cake sweet but their drink with no sugar, the attribute-injected encoder would give a positive signal to the encoding of sweet in the text “the cake was sweet” and a negative signal in the text “the drink was sweet”.

How and Where to Inject Attributes? ::: Where: Attribute Injection ::: In the logistic classifier

Injecting attributes to the classifier means that we bias the probability distribution of sentiment based on the final document encoding. If a user tends to classify the sentiment of reviews about sweet cakes as highly positive, then the model would give a high probability to highly positive sentiment classes for texts such as “the cake was sweet”.

Experiments ::: General Setup

We perform experiments on two tasks. The first task is Sentiment Classification, where we are tasked to classify the sentiment of a review text, given additionally the user and product information as attributes. The second task is Attribute Transfer, where we attempt to transfer the attribute encodings learned from the sentiment classification model to solve two other different tasks: (a) Product Category Classification, where we are tasked to classify the category of the product, and (b) Review Headline Generation, where we are tasked to generate the title of the review, given only both the user and product attribute encodings. Datasets, evaluation metrics, and competing models are different for each task and are described in their corresponding sections.

Unless otherwise stated, our models are implemented with the following settings. We set the dimensions of the word, user, and product vectors to 300. We use pre-trained GloVe embeddings BIBREF28 to initialize the word vectors. We also set the dimensions of the hidden state of BiLSTM to 300 (i.e., 150 dimensions for each of the forward/backward hidden state). The chunk size factors C_1 and C_2 are both set to 15. We use dropout BIBREF29 on all non-linear connections with a dropout rate of 0.1. We set the batch size to 32. Training is done via stochastic gradient descent over shuffled mini-batches with the Adadelta update rule BIBREF30 and with l_2 constraint BIBREF31 of 3. We perform early stopping using the development set. Training and experiments are done using an NVIDIA GeForce GTX 1080 Ti graphics card.

Experiments ::: Sentiment Classification ::: Datasets and Evaluation

We use the three widely used sentiment classification datasets with user and product information available: IMDB, Yelp 2013, and Yelp 2014 datasets. These datasets are curated by BIBREF12, where they ensured twenty-core for both users and products (i.e., users have at least twenty products and vice versa), split them into train, dev, and test sets with an 8:1:1 ratio, and tokenized and sentence-split using the Stanford CoreNLP BIBREF32. Dataset statistics are shown in Table TABREF20. Evaluation is done using two metrics: the accuracy which measures the overall sentiment classification performance, and RMSE which measures the divergence between predicted and ground truth classes.

Experiments ::: Sentiment Classification ::: Comparisons of different attribute representation and injection methods

To conduct a fair comparison among the different methods described in Section SECREF2, we compare these methods when applied to our base model using the development set of the datasets. Specifically, we use a smaller version of our base model (with dimensions set to 64) and incorporate the user and product attributes using nine different approaches: (1) bias-attention: the bias-based method injected to the attention mechanism, (2-5) the matrix-based method injected to four different locations (matrix-embedding, matrix-encoder, matrix-attention, matrix-classifier), and (6-9) the CHIM-based method injected to four different locations (CHIM-embedding, CHIM-encoder, CHIM-attention, CHIM-classifier). We then calculate the accuracy of each approach for all datasets.

Results are shown in Figure FIGREF25. The figure shows that bias-attention consistently performs poorly compared to other approaches. As expected, matrix-based representations perform the worst when injected to embeddings and encoder, however we can already see improvements over bias-attention when these representations are injected to attention and classifier. This is because the number of parameters used in the the weight matrices of attention and classifier are relatively smaller compared to those of embeddings and encoder, thus they are easier to optimize. The CHIM-based representations perform the best among other approaches, where CHIM-embedding garners the highest accuracy across datasets. Finally, even when using a better representation method, CHIM-attention consistently performs the worst among CHIM-based representations. This shows that attention mechanism is not the optimal location to inject attributes.

Experiments ::: Sentiment Classification ::: Comparisons with models in the literature

We also compare with models from previous work, listed below:

UPNN BIBREF12 uses a CNN classifier as base model and incorporates attributes as user- and product-specific weight parameters in the word embeddings and logistic classifier.

UPDMN BIBREF16 uses an LSTM classifier as base model and incorporates attributes as a separate deep memory network that uses other related documents as memory.

NSC BIBREF7 uses a hierarchical LSTM classifier as base model and incorporates attributes using the bias-attention method on both word- and sentence-level LSTMs.

DUPMN BIBREF17 also uses a hierarchical LSTM as base model and incorporates attributes as two separate deep memory network, one for each attribute.

PMA BIBREF14 is similar to NSC but uses external features such as the ranking preference method of a specific user.

HCSC BIBREF9 uses a combination of BiLSTM and CNN as base model, incorporates attributes using the bias-attention method, and also considers the existence of cold start entities.

CMA BIBREF15 uses a combination of LSTM and hierarchical attention classifier as base model, incorporates attributes using the bias-attention method, and does this separately for user and product.

Notice that most of these models, especially the later ones, use the bias-attention method to represent and inject attributes, but also employ a more complex model architecture to enjoy a boost in performance. Results are summarized in Table TABREF33. On all three datasets, our best results outperform all previous models based on accuracy and RMSE. Among our four models, CHIM-embedding performs the best in terms of accuracy, with performance increases of 2.4%, 1.3%, and 1.6% on IMDB, Yelp 2013, and Yelp 2014, respectively. CHIM-classifier performs the best in terms of RMSE, outperforming all other models on both Yelp 2013 and 2014 datasets. Among our models, CHIM-attention mechanism performs the worst, which shows similar results to our previous experiment (see Figure FIGREF25). We emphasize that our models use a simple BiLSTM as base model, and extensions to the base model (e.g., using multiple hierarchical LSTMs as in BIBREF21), as well as to other aspects (e.g., consideration of cold-start entities as in BIBREF9), are orthogonal to our proposed attribute representation and injection method. Thus, we expect a further increase in performance when these extensions are done.

Experiments ::: Attribute Transfer

In this section, we investigate whether it is possible to transfer the attribute encodings, learned from the sentiment classification model, to other tasks: product category classification and review headline generation. The experimental setup is as follows. First, we train a sentiment classification model using an attribute representation and injection method of choice to learn the attribute encodings. Then, we use these fixed encodings as input to the task-specific model.

Experiments ::: Attribute Transfer ::: Dataset

We collected a new dataset from Amazon, which includes the product category and the review headline, aside from the review text, the sentiment score, and the user and product attributes. Following BIBREF12, we ensured that both users and products are twenty-core, split them into train, dev, and test sets with an 8:1:1 ratio, and tokenized and sentence-split the text using Stanford CoreNLP BIBREF32. The final dataset contains 77,028 data points, with 1,728 users and 1,890 products. This is used as the sentiment classification dataset. To create the task-specific datasets, we split the dataset again such that no users and no products are seen in at least two different splits. That is, if user u is found in the train set, then it should not be found in the dev and the test sets. We remove the user-product pairs that do not satistfy this condition. We then append the corresponding product category and review headline for each user-product pair. The final split contains 46,151 training, 711 development, and 840 test instances. It also contains two product categories: Music and Video DVD. The review headline is tokenized using

SentencePiece with 10k vocabulary. The datasets are released here for reproducibility: https://github.com/rktamplayo/CHIM.

Experiments ::: Attribute Transfer ::: Evaluation

In this experiment, we compare five different attribute representation and injection methods: (1) the bias-attention method, and (2-5) the CHIM-based representation method injected to all four different locations in the model. We use the attribute encodings, which are learned from pre-training on the sentiment classification dataset, as input to the transfer tasks, in which they are fixed and not updated during training. As a baseline, we also show results when using encodings of randomly set weights. Moreover, we additionally show the majority class as additional baseline for product category classification. For the product category classification task, we use a logistic classifier as the classification model and accuracy as the evaluation metric. For the review headline generation task, we use an LSTM decoder as the generation model and perplexity as the evaluation metric.

Experiments ::: Attribute Transfer ::: Results

For the product category classification task, the results are reported in Table TABREF47. The table shows that representations learned from CHIM-based methods perform better than the random baseline. The best model, CHIM-encoder, achieves an increase of at least 3 points in accuracy compared to the baseline. This means that, interestingly, CHIM-based attribute representations have also learned information about the category of the product. In contrast, representations learned from the bias-attention method are not able to transfer well on this task, leading to worse results compared to the random and majority baseline. Moreover, CHIM-attention performs the worst among CHIM-based models, which further shows the ineffectiveness of injecting attributes to the attention mechanism.

Results for the review headline generation task are also shown in Table TABREF47. The table shows less promising results, where the best model, CHIM-encoder, achieves a decrease of 0.88 points in perplexity from the random encodings. Although this still means that some information has been transferred, one may argue that the gain is too small to be considered significant. However, it has been well perceived, that using only the user and product attributes to generate text is unreasonable, since we expect the model to generate coherent texts using only two vectors. This impossibility is also reported by BIBREF8 where they also used sentiment information, and BIBREF33 where they additionally used learned aspects and a short version of the text to be able to generate well-formed texts. Nevertheless, the results in this experiment agree to the results above regarding injecting attributes to the attention mechanism; bias-attention performs worse than the random baseline, and CHIM-attention performs the worst among CHIM-based models.

Experiments ::: Where should attributes be injected?

All our experiments unanimously show that (a) the bias-based attribute representation method is not the most optimal method, and (b) injecting attributes in the attention mechanism results to the worst performance among all locations in the model, regardless of the representation method used. The question “where is the best location to inject attributes?” remains unanswered, since different tasks and settings produce different best models. That is, CHIM-embedding achieves the best accuracy while CHIM-classifier achieves the best RMSE on sentiment classification. Moreover, CHIM-encoder produces the most transferable attribute encoding for both product category classification and review headline generation. The suggestion then is to conduct experiments on all locations and check which one is best for the task at hand.

Finally, we also investigate whether injecting in to more than one location would result to better performance. Specifically, we jointly inject in two different locations at once using CHIM, and do this for all possible pairs of locations. We use the smaller version of our base model and calculate the accuracies of different models using the development set of the Yelp 2013 dataset. Figure FIGREF49 shows a heatmap of the accuracies of jointly injected models, as well as singly injected models. Overall, the results are mixed and can be summarized into two statements. Firstly, injecting on the embedding and another location (aside from the attention mechanism) leads to a slight decrease in performance. Secondly and interestingly, injecting on the attention mechanism and another location always leads to the highest increase in performance, where CHIM-attention+embedding performs the best, outperforming CHIM-embedding. This shows that injecting in different locations might capture different information, and we leave this investigation for future work.

Related Work ::: Attributes for Sentiment Classification

Aside from user and product information, other attributes have been used for sentiment classification. Location-based BIBREF34 and time-based BIBREF35 attributes help contextualize the sentiment geographically and temporally. Latent attributes that are learned from another model have also been employed as additional features, such as latent topics from a topic model BIBREF36, latent aspects from an aspect extraction model BIBREF37, argumentation features BIBREF38, among others. Unfortunately, current benchmark datasets do not include these attributes, thus it is practically impossible to compare and use these attributes in our experiments. Nevertheless, the methods in this paper are not limited to only user and product attributes, but also to these other attributes as well, whenever available.

Related Work ::: User/Product Attributes for NLP Tasks

Incorporating user and product attributes to NLP models makes them more personalized and thus user satisfaction can be increased BIBREF39. Examples of other NLP tasks that use these attributes are text classification BIBREF27, language modeling BIBREF26, text generation BIBREF8, BIBREF33, review summarization BIBREF40, machine translation BIBREF41, and dialogue response generation BIBREF42. On these tasks, the usage of the bias-attention method is frequent since it is trivially easy and there have been no attempts to investigate different possible methods for attribute representation and injection. We expect this paper to serve as the first investigatory paper that contradicts to the positive results previous work have seen from the bias-attention method.

Conclusions

We showed that the current accepted standard for attribute representation and injection, i.e. bias-attention, which incorporates attributes as additional biases in the attention mechanism, is the least effective method. We proposed to represent attributes as chunk-wise importance weight matrices (CHIM) and showed that this representation method significantly outperforms the bias-attention method. Despite using a simple BiLSTM classifier as base model, CHIM significantly outperforms the current state-of-the-art models, even when those models use a more complex base model architecture. Furthermore, we conducted several experiments that conclude that injection to the attention mechanism, no matter which representation method is used, garners the worst performance. This result contradicts previously reported conclusions regarding attribute injection to the attention mechanism. Finally, we show promising results on transferring the attribute representations from sentiment classification, and use them to two different tasks such as product category classification and review headline generation.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful feedback and suggestions. Reinald Kim Amplayo is grateful to be supported by a Google PhD Fellowship.