CSE 576 Natural Language Processing
Project Third (Final) Phase Report
Aspect Based Sentiment Analysis

1207030067, 1212335928, 1208330509, 1203043551, 1210467138
Skyler Rothman, **Sai Shi**, **Mark Strickland**, **Dian Xu**, **Haisi Yi**
Arizona State University, Arizona
{sjrothma, sshi23, mestric1, dianxu, haisiyi}@asu.edu

1 Introduction & Motivation

As an opinion mining technique, Aspect Based Sentiment Analysis (ABSA) offers an effective solution for extracting fine-grained sentiment information from the overwhelming amount of text on the Internet. ABSA has very promising potential applications in product improvement and market analysis. However, sentiment analysis in NLP is a challenging topic and ABSA elevates the challenge by focusing on predictions relating to specified aspects of the text. The current models using neural networks can only achieve an accuracy slightly over 80% in three-way classification (positive, neutral and negative), which still leaves much room for improvement.

One problem that adversely affects the performance of sentiment analysis is the length of the input text. Long sentences may make it more difficult to identify the relevant words in the text related to the target aspect, as well as words relevant to the polarity of the sentiment being expressed, and thereby reduce the accuracy of the analysis. Another problem having a negative effect on sentiment analysis performance relates to the defined aspect classes themselves. If the aspect classes are ill-defined, for instance the class definitions use similar, complex or obscure words, this may present problems, e.g., relating to dictionary embeddings of the words, and as a result sentiment analysis may suffer. To avoid these issues, various solutions have been attempted on an individual basis, including the use of multiple embedding dictionaries, variation of parameters for LSTM layers, attention mechanisms, etc.

In this project, we propose to make improvements on ABSA through a combination of building more complex neural networks using LSTM (long short-term memory) layers, attention mechanisms and additional feature engineering.

The input provided to our model will consist of one or more review texts, each including: (a) one or more sentences, each sentence consisting of a sequence of words; and (b) a tuple of labels, each label containing the target aspect identifier and its corresponding polarity (e.g. positive, negative, or neutral) in the sentence.

The aspect identifier portion of the label varies according to dataset. For the SemEval2014 dataset, the aspect identifier is a word or phrase identifying the aspect being reviewed in the sentence (e.g., "Price"). For the SemEval2016 dataset, the aspect identifier is an entity # attribute pair that identifies the aspect category and subcategory being reviewed in the sentence (e.g., Food # Quality).

As should be obvious, the review text sentences (a) are the portion of the input used by our model to generate the prediction of polarity - the label is used to facilitate network learning in the typical supervised learning process. For simplicity, our model's output will consist of a single polarity per sentence, as more formally defined below.

More formally, the SemEval2016 ABSA task requires an output in the form S:

$$S = (E\#A, P)$$

where E is the entity designating the aspect category and A is the attribute designating the aspect subcategory. P is the polarity of the review, i.e., positive, negative, or neutral.

The SemEval2014 ABSA task, on the other hand, employs a simplified form:

$$S' = (A, P)$$

where A is the aspect identifier and P is the polarity.

Our model, in contrast, will perform a subset of the SemEval2014 ABSA task, and will simply predict the polarity of an input sentence, for a given aspect:

$$S'' = (P)$$

where P is the polarity of the sentence with respect to a given aspect.

For the SemEval2014 dataset, this simplification will require few or no changes to the input data. However, for the SemEval2016 dataset, we will need to simplify the input labels. For example, we may reduce each entity/attribute pair to a single aspect identifier by dropping the attribute, which is simply a subcategory in most cases.

1.1 Input

As noted, the input to our model consists of individual sentences, each of which may be defined as a sequence of words. Our model converts each word to a numerical vector, which is assembled together with the other words in the sentence into a convenient matrix form. The expression of each word as a vector is handled using the popular "GloVe" word embedding technique. (Pennington et al., 2014) In particular, our model uses a 300-dimensional feature vector to represent each word. More formally, the input to our model is (per example):

$$input = (w_1, w_2, ...w_N)$$

where w_i are individual words in the overall sequence and N is the number of words in the sentence. Each of the words w_i is transformed by our model into a (300 x 1) vector using the "GloVe" embedding.

1.2 Output

The output of our model will include the polarity, as defined above for a given aspect. More formally, the output provided by our model will consist of (per example):

$$given : aspect$$

$$output = (polarity)$$

where $aspect$ is an integer corresponding to one of several predetermined aspect classes (e.g., 0=food, 1=service, 2=price, 3=ambience, 4=miscellaneous) which may be translated back into a word or phrase denoting the class for human-readability, and $polarity$ is an integer corresponding to one of several predetermined polarity classes (e.g., +1=positive, 0=neutral, -1=negative).

2 Related Work

As one of the SemEval challenges, ABSA was first introduced in 2014. (Pavlopoulos, 2014), (Zhang and Liu, 2014) Even before 2014, Support Vector Machines (SVM) with target-dependent and target-independent feature-engineering methods were applied and proven to be effective. (Moraes et al., 2013) Many approaches have been applied to solve this task since then.

Neural network approaches were used and achieved better performance compared to SVM. Among the neural network models that have been applied, recurrent neural networks and specifically the long short-term memory (LSTM) model is the most popular and promising one. In 2015, Tang et al. proposed TD-LSTM and TC-LSTM models to force the network to better remember the aspect information in the input and achieved significant improvements in performance. (Tang et al., 2015) In 2016, Wang et al. added an attention mechanism on top of LSTM and learned an aspect matrix to represent the aspect information, achieving even better performance. (Wang et al., 2016) Recently in 2017, on top of the model using LSTM and attention, Chen et al. proposed a Recurrent Attention network on Memory (RAM) model where they added a position-weighted memory module, and used a recurrent GRU to

distill information from the memory, and obtained the best performance so far. (Chen et al., 2017)

There are certain problems related to the most recent two models. First, minimum preprocessing was done on the raw input which leads to about 5% out-of-vocabulary words even when using the 840B GloVe dictionary. The unknown words may contain key information such as aspect and sentiment polarities. The ATAE model proposed by Wang et al. used only a simple LSTM as the backbone of the architecture. (Wang et al., 2016) Though the inclusion of aspect embedding and attention mechanism added complexity to the model, important information relating to full sentence encoding was missed. On the contrary, the RAM model used a very complex model with a bidirectional LSTM (BiLSTM) as the backbone. (Chen et al., 2017) The most important information used as input was the aspect terms (words or phrases related to the aspects) to apply weights on the cell memory from BiLSTM. Though the aspect term information is available for use as defined in SemEval, it is not easily available for other datasets, therefore limiting the applicability of the model. In our project, we added improvements to the current models. Using the ATAE model as the main baseline, we added: (a) heavy text preprocessing to minimize the number of out-of-vocabulary words while preserving the original meanings of the words, and (b) BiLSTM layers with bidirectional aspect embeddings.

3 The Approach

3.1 Intuition of the Approach

The algorithms being used in our project can be split into two phases. The first phase is related to preprocessing the raw text data available in the xml format. The second phase is a neural network architecture that takes the input from phase one to form a generalized model for aspect-level sentiment analysis.

For preprocessing (Figure 1), the data of the training and testing datasets was parsed from XML to extract the sentences and corresponding aspect category with polarities. During the preprocessing, for each sentence that contains multiple aspect polarities, it was duplicated for each aspect polarity. Meanwhile, all instances with "conflict" polarities were filtered out. Next, all sentences were split into words using nltk's word tokenizer. The resulting word list was reduced to form the comprehensive set of unique words to be candidates for the vocabulary of the model instance. Words in the set that can be found in the GloVe dictionary were added to the vocabulary, while for the rest of the unrecognized words, the following procedures were done on them:

If a word contains dash, split it based on dash, and for each of the resulting words, check its spelling using hunspell and for misspelled words, use hunspell and edit_distance from nltk to correct its spelling. The resulting words were reconstructed with dash and looked up in GloVe. If the reconstructed words were still missing in GloVe, the split words will be added to the vocabulary.
If a word is a numeric word that expresses certain time in a day, e.g. "6:05pm", it is converted into a corresponding English word describing the corresponding time, e.g. "evening".
If a word contains "+" symbol or numbers that are next to other English words, the word is split into symbol or numbers and the following word.
If a word does not fall into any of the situations above, it is labeled as 'unknown' and represented as a randomized vector using normal distribution between -1 and 1.

The resulting vocabulary contained all unique words appearing in the training and testing datasets, and each of the words was represented by the corresponding vectors from GloVe. The vocabulary served as a lookup table for the neural network model to use as input. After the preprocessing, the number of out-of-vocabulary words was largely reduced from 230 to only 5 words.

Regarding the second phase of our project algorithms, we use a BiLSTM as the encoder for the sentences and use two separate aspect embedding matrix to concatenate with the hidden layer outputs from the BiLSTM. On top of the BiLSTM and aspect embedding, we used an attention layer to learn weights from each hidden state of the LSTM. As noted in (Chen et al., 2017), the update process for each of bi-directional LSTMs is given by:

$$i = \sigma(\vec{W}_i \vec{h}_t^{l-1} + \vec{U}_i \vec{h}_{t-1}^{l})$$

$$f = \sigma(\vec{W}_f \vec{h}_t^{l-1} + \vec{U}_f \vec{h}_{t-1}^{l})$$

$$o = \sigma(\vec{W}_o \vec{h}_t^{l-1} + \vec{U}_o \vec{h}_{t-1}^{l})$$

$$g = tanh(\vec{W}_g \vec{h}_t^{l-1} + \vec{U}_g \vec{h}_{t-1}^{l})$$

$$\vec{c_t}^{l} = f \odot \vec{c}_{t-1}^{l} + i \odot g$$

$$\vec{h_t}^{l} = o \odot tanh(\vec{c_t}^{l})$$

where, at each time step t, the LSTM outputs the hidden state $\vec{h}_t^l$ and additionally maintains a memory $\vec{c_t}^l$ inside its hidden cell.

The inputs into the BiLSTM were from the GloVe 6B word embedding dictionary. After the word vector of each word of the sentence is fed into the BiLSTM, the hidden states from the forward LSTM is concatenated with the corresponding aspect vector in the forward aspect matrix. The backward LSTM is doing the same thing with the backward aspect vectors. The concatenated matrix H is fed into the attention layer where a weighting score alpha is learned through training by applying the tanh and softmax layers. After this, H is weighted by the alpha to form the attended encoded sentence information. The steps can be described by the following equations:

$$M = tanh\left(\begin{bmatrix} W_h H \\ W_v v_a \otimes e_N \end{bmatrix}\right)$$

$$\alpha = softmax(w^T M)$$

$$r = H \cdot \alpha_T$$

where: $M \in \mathbb{R}^{(2d+2d_a)\times N}$, $\alpha \in \mathbb{R}^N$, $r \in \mathbb{R}^d$, H is the matrix of hidden vectors $(h_1, ..., h_N)$ that the LSTM produced, $W_h \in \mathbb{R}^{2d\times 2d}$, $W_v \in \mathbb{R}^{2d_a \times 2d_a}$, and $w \in \mathbb{R}^{(2d+2d_a)}$.

Note that $v_a \otimes e_N = [v; v; ...; v]$, or the operator $\otimes$ repeatedly concatenates v for N times, where e_N is a column vector with N ones.

The final sentence representation is given by:

$$h^* = tanh(W_p r + W_x h_N)$$

where W_p and W_x are projection parameters learned during training.

The network output is given by:

$$y = softmax(W_s h^* + b_s)$$

where W_s and b_s are the parameters of the softmax layer.

The network output y is a one-hot vector indicating the predicted polarity with respect to the specified aspect.

Our network architecture, in contrast, adds bi-directional LSTM layers (one LSTM layer for the forward direction, and another LSTM for the backward direction) with attention. Referring to Figure 1, the input to our network is the word vectors and the aspect embeddings, and the output is the polarity predicted with respect to that aspect.

3.2 Training

The network is trained end-to-end using backpropagation, where the objective function is cross-entropy loss. The goal of training is to minimize the cross-entropy loss between y and $\hat{y}$:

$$loss = -\Sigma_i \Sigma_j y_i{}^j log(\hat{y}_i{}^j) + \lambda \|\theta\|^2$$

where y_i is the target (label) distribution for sentence i, $\hat{y}_i$ is the predicted sentiment distribution for sentence i, j is the index of the

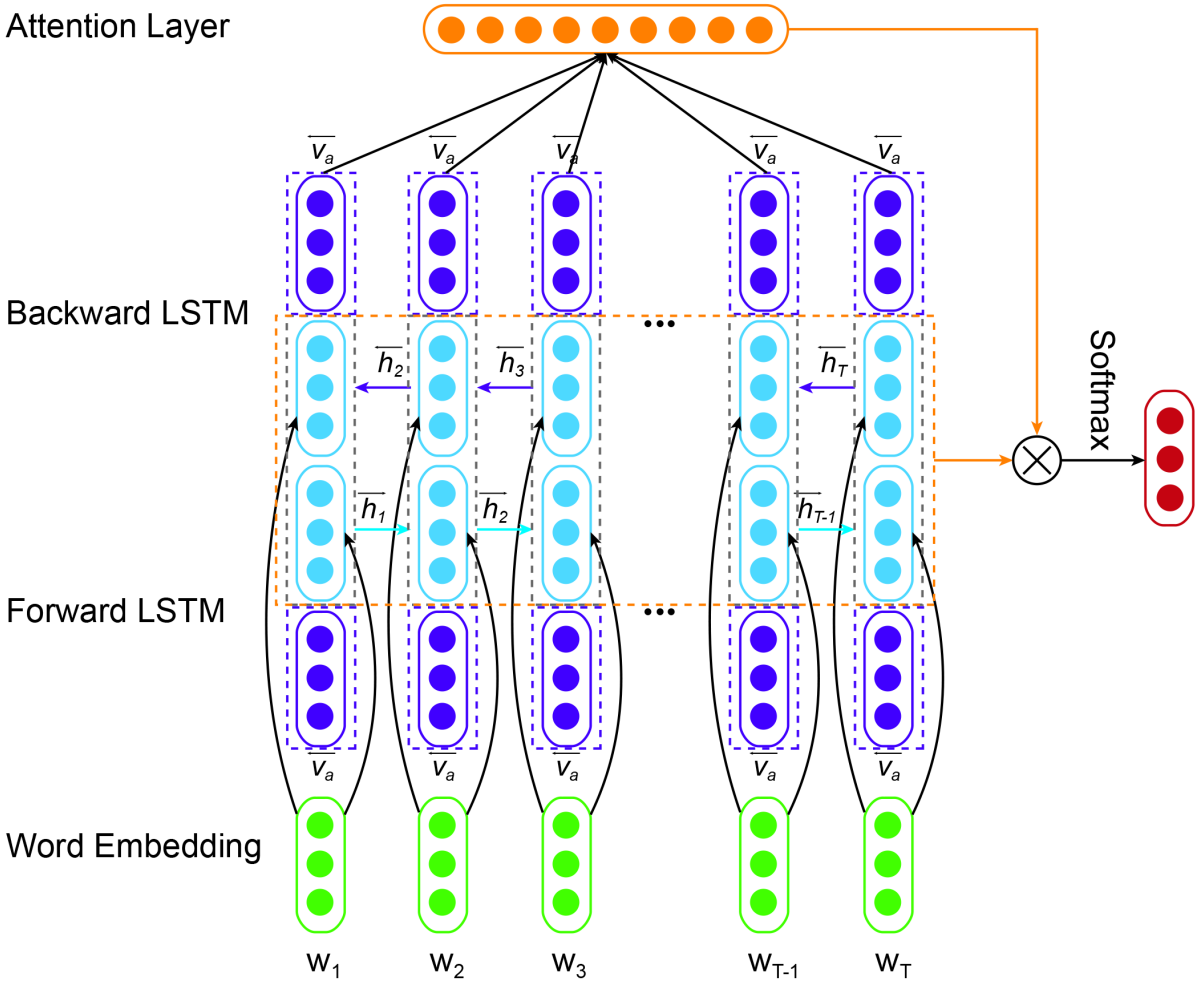

Figure 1: Our Network Architecture, including Bi-directional Attention-based LSTMs with Aspect Embedding.

polarity class (positive, negative, or neutral). λ is a regularization term and θ is the parameter set learned during training (e.g., $\{W_i, b_i, W_f, b_f, W_o, b_o, W_c, b_c, W_s, b_s, W_h, W_v, W_p, W_x\}$).

We used the ADAM optimizer with an initial learning rate of 0.01 for lower dimensions (n_lstm = 50 and 100) lstm and 0.001 for higher dimensions (n_lstm = 200 and 300) in LSTM layer. (Duchi et al., 2011) This optimizer adapts the learning rate to the parameters of the particular optimization problem being solved, performing larger updates for certain parameters and smaller updates for other parameters.

3.3 Training Dataset

The existing datasets for ABSA include the data from SemEval-2014 Task 4 and SemEval-2016 Task 5. Only the English version of these datasets will be used in this project. Both training datasets are decently sized (SemEval-2014 dataset includes 3041 sentences related to restaurants and 3045 sentences relating to laptops; SemEval-2016 dataset includes 2000 sentences related to restaurants and 2500 sentences related to laptops) and include gold-standard annotations. A statistics of the dataset is shown in Figure 3.

Our preprocessing code takes as input the original .xml files, e.g., released with SemEval-2014 Task4, parses the sentences, tokenizes them, and performs corrections, as described above (see Section 4, Inference and Figure 1).

3.4 Example

An example input that we could classify is a sentence "The staff at Dilly Diner were friendly and very prompt." After preprocessing, each token "the", "staff", ..., "prompt", "." is individually embedded into vector form. The aspect to be queried for this sentence could be "service", which is also embedded into vector form. The output of the model would reveal the most likely polarity of the given sentence with respect to the queried polarity. For this example sentence, since it is describing the staff of the restaurant as friendly and prompt, we should expect the model to return the polarity as "1" to denote that the sentence describes the service of the restaurant positively.

4 Evaluation and Results

Since we are inputting each sentence along with its respective aspect for sentiment analysis to calculate the polarity with respect to the target, the accuracy of the polarity is evaluated. For each entry in the test data, the correct polarity is known and is compared to the polarity that the model calculates. We will focus on the three-way classification, such that the model outputs positive (1), negative (-1), or neutral (0) for every sentence in the test data, and when it is compared to the correct polarity, output accuracy is measured based simply on whether or not it achieved the correct polarity. (e.g. If the correct polarity is -1, there is no difference in "wrongness" between the model determining 0 or determining 1.) The overall accuracy is simply the number of correct polarity determinations divided by the number of test sentences.

To optimize the hyperparameters of our model, systematic experiments were conducted to find the optimal results. The main parameters that define the hyperplane of the loss function of the model are the dimensions of lstm (n_lstm) and aspect embeddings (n_AE). We selected 50, 100, 200 and 300 four points for each of the parameter and formed a total of 16 permutations. The training loss/accuracy and testing loss/accuracy were plotted in Figure 2. During optimization, the learning rate was adjusted to help the model find the mininum. As in a simpler model (lower n_lstm and n_AE), the hyperplane of the loss function is simpler and requires larger learning rate to reach the mininum faster while for more complex model with higher dimensionalities, smaller learning rate can help the model to find the mininum better. A total of 100 iterations were done for all testing cases. Except for the 100 (n_lstm), 200 (n_AE), all permutations showed process through training process. And the optimal testing accuracy was found to be 50 (n_lstm), 300 (n_AE). When compared with the basic LSTM model with 300 n_lstm value, out model still outperform it by 2%. Compare with the ATAE model, our model is 3.3% better.

The effects of n_lstm and n_AE were shown in Figure 4. When n_lstm is 50 or 300, the performances are better than those of n_lstm = 100 and 200. Also, for dimensions of aspect embedding,

the highest value of 300 is generally better than the lower values. Our hyperparameter searching

5 Error Analysis

Using the trained model, we reviewed several of the cases where a sentence's polarity was incorrectly classified.

The model often incorrectly classifies polarities when there are two or more aspects in the same sentence that have different polarities, even when the structure of the sentence is simple and the expected polarities for certain aspects are explicit. For example, in the following two sentences, the sentences were incorrectly classified with a negative polarity with respect to food:

"The food was delicious, but it was too expensive."

"The food was great, but the service was quite poor."

The same effect was noticed with the polarity flipped, as well. In the following two sentences, the sentences were incorrectly classified with a positive polarity with respect to food:

"The food was bland, but it was very cheap."

"The food was flavorless, but the service was excellent."

Interestingly, two of the prior sentences got the correct polarity with identical wording, except the order of the clauses are swapped. The following two sentences get the polarity of food correct as positive and negative respectively:

"The service was quite poor, but the food was great."

"it was very cheap, but the food was bland. "

Despite the semantics of the above sentence being identical, the model classifies these sentences differently depending on the order of the clauses. This indicates that the model may be associating polarities differently depending on if certain words are towards the end of the sentence versus the beginning. Ideally, the bi-directional LSTM should alleviate some of this effect compared to a single-direction LSTM, but it may not be the perfect solution for this particular kind of case.

The model also seems to struggle with appropriately classifying sentences that should be classified as neutral. This was especially common when other aspects/adjectives with a non-neutral polarity were present. For example, the following incorrectly sentences classified the polarity as positive with respect to food when a neutral statement regarding food is made:

"it was very cheap, and I got a shrimp soup."

"The service was excellent, and there was a big vat of soup. "

This could be in part due common words being used to describe positive or negative sentiments, but the range of words used to describe a neutral statement is wide and not well-defined.

For some sentences it appears the model can be tricked by words that clearly convey one polarity on their own, but are used to convey the opposite polarity when given more context. The following sentence is classified as positive with respect to service, but is actually negative:

"The staff should be a bit more friendly"

This sort of case did not seem to be an issue when the negation of polarity in a simple way, such as using the word "not".

The model seems to have a difficult time classifying idioms, metaphors, or language that is not explicit in the expected polarity. Here are some examples of sentences that were misclassified:

"The two waitresses looked like they had been sucking on lemons."

"If you're in new york , you do not want to miss this place."

"however , it was worth the visit."

Many of the errors could be placed in one of the above cases, but some sentences are misclassified without an easy explanation. It would take a more in-depth analysis of the model's encoding of the sentence to get a clear understanding of why it is not classifying them with the correct polarity.

6 Conclusion & Future Work

In sum, our project resulted in a novel model architecture that outperformed the baselines for the SemEval-2014 restaurant review dataset for sentences. Our accuracy results were competitive with the state-of-the-art for Aspect Based Sentiment Analysis, for 3-way polarity prediction. Our hyperparameter search revealed that increasing

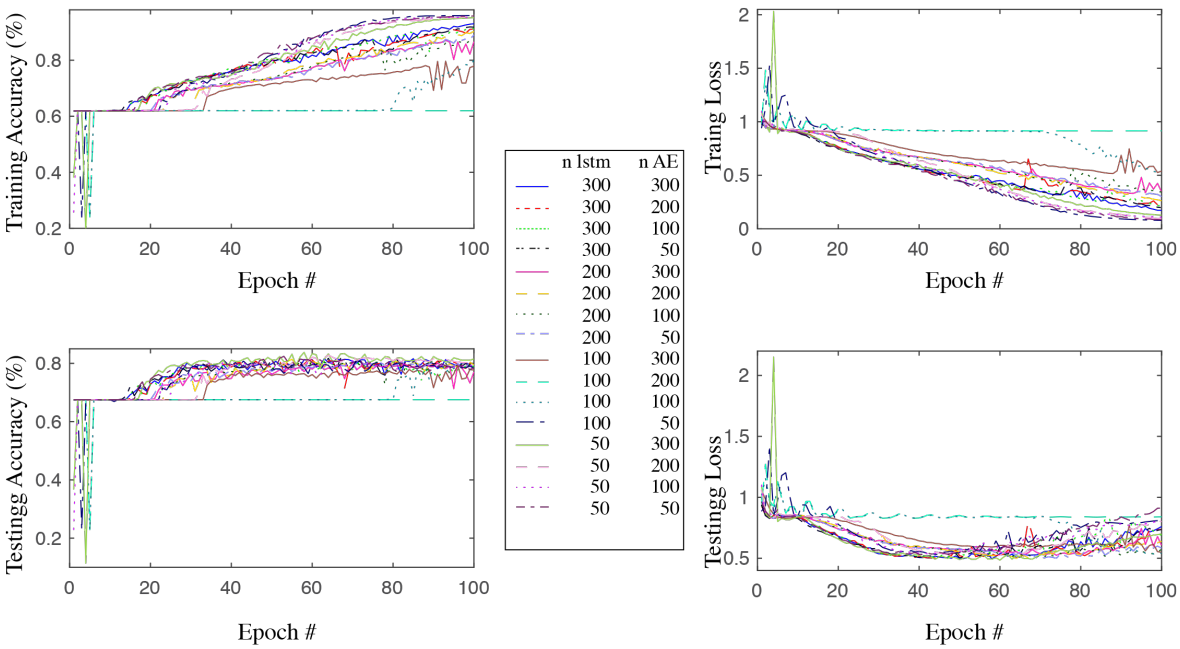

Figure 2: Training accuracy and loss curves for our model, comparing various combinations of hyperparameters. n_lstm refers the dimensionality of the LSTM layers and n_AE refers to the dimensionality of the aspect embedding.

Aspect	Positive		Neutral		Negative	
	Train	Test	Train	Test	Train	Test
Fo.	867	302	209	69	90	31
Pr.	179	51	115	28	10	1
Se.	324	101	218	63	20	3
Am.	263	76	98	21	23	8
Mi.	546	127	199	41	357	51
Total	2179	657	839	222	500	94

Figure 3: Description of the dataset used to train and test our model, taken from SemEval-2014 Task 4, showing the number of example restaurant reviews for each of the 5 aspect classes and 3-way polarity labels.

n lstm \ n AE	50	100	200	300
50	82.4	81.0	82.8	83.8
100	81.7	80.0	67.5	79.3
200	80.7	81.1	81.4	80.2
300	81.8	80.9	81.2	81.6

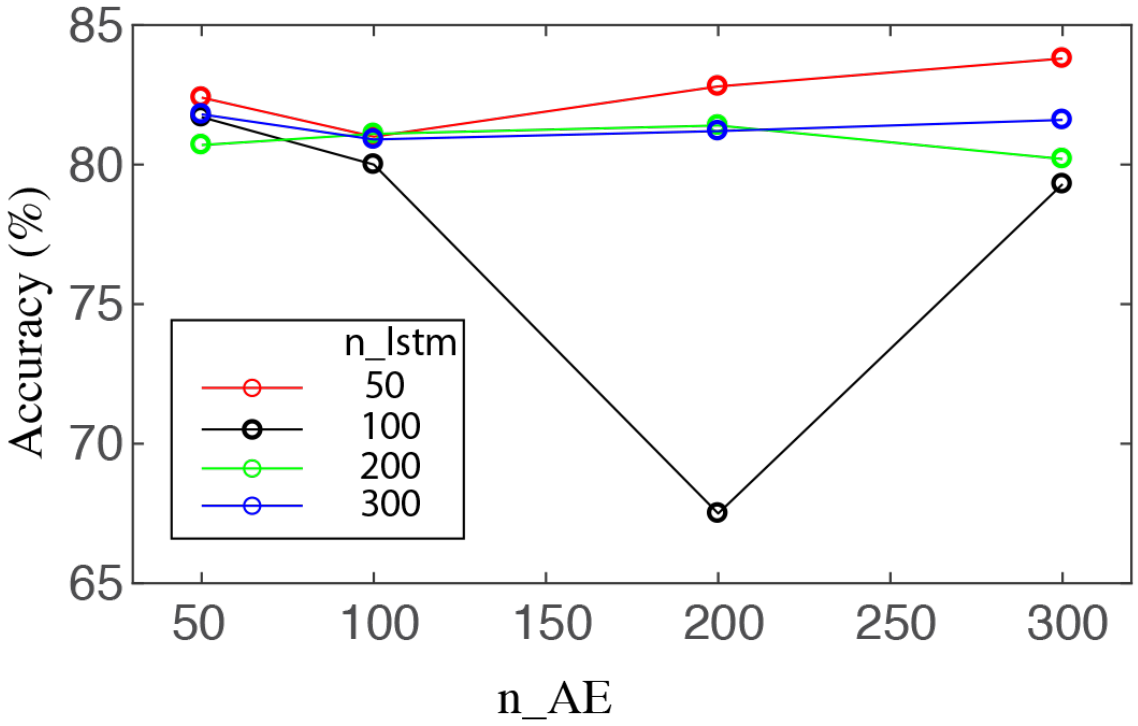

Figure 4: Hyperparameter search space and resulting test accuracies for each hyperparameter combination. n_lstm refers the dimensionality of the LSTM layers and n_AE refers to the dimensionality of the aspect embedding.

Models	LSTM	BiLSTM	ATAE	**This paper**
Accuracy	81.8	80.3	80.5	**83.8**

Figure 5: Accuracy comparison with baselines. Note that our model outperformed all baselines for the SemEval-2014 sentence-level restaurant review dataset.

dimensionality in some areas was more productive than in others, adding insights about the nature of using attention-based bi-directional LSTM layers for sentiment analysis. Finally, our training curves revealed an upper limit on the training necessary for this type of model.

One area in which future work should be performed is the testing of our network on other datasets. Although we analyzed the SemEval-2016 Task5 data, we did not perform any training or testing of our network (or any of the baselines) with the SemEval-2016 data. This is primarily because of the difference in aspect identification in this dataset. While the SemEval-2014 data included a single set of aspect classes (i.e., each example sentence was assigned to one aspect class), the SemEval-2016 data utilized aspect subclasses, so that each example was assigned a higher-level aspect and a lower-level subaspect. In addition, for the restaurant reviews we had been analyzing, the aspect classes and subclasses in the SemEval-2016 data were less well-defined and both overlapped with the SemEval2014 aspect classes. In the end, we decided to concentrate our efforts in training and testing on the SemEval-2014 data. However, a natural extension of our project would be to adapt the SemEval-2016 data for training and testing.

Another area of future work could be comparison with additional baselines. As noted above, other model types have been used for sentiment analysis in the past, including Bag-of-Words, SVMs, etc. Future work could include adapting these other model types to perform the similar ABSA task as our model, for comparison purposes. However, this is likely to be a strictly academic exercise, since the performance of recurrent networks has already been seen to outperform these other, prior model types for less specific (non-aspect-based) sentiment analysis.

References

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 452–461.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Rodrigo Moraes, JoãO Francisco Valiati, and Wilson P GaviãO Neto. 2013. Document-level sentiment classification: An empirical comparison between svm and ann. *Expert Systems with Applications*, 40(2):621–633.

Ioannis Pavlopoulos. 2014. Aspect based sentiment analysis. *Athens University of Economics and Business*.

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

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Effective lstms for target-dependent sentiment classification. *arXiv preprint arXiv:1512.01100*.

Yequan Wang, Minlie Huang, xiaoyan zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, Austin, Texas, November. Association for Computational Linguistics.

Lei Zhang and Bing Liu. 2014. Aspect and entity extraction for opinion mining. In *Data mining and knowledge discovery for big data*, pages 1–40. Springer.