DEPARTMENT OF INFORMATICS
M.Sc. in COMPUTER SCIENCE

Master of Science Thesis in

Aspect-Based Sentiment Analysis for restaurants from reviews in the English language

Admir Demiraj

EY1613

Supervisor: Ion Androutsopoulos

Athens, June 2018

Acknowledgements

I would like to thank first and above all my supervisor Prof. Ion Androutsopoulos for giving me the opportunity to work in this exciting field and for his guidance throughout the tasks of this thesis. Moreover, I would like to thank Christos Baziotis and the other members of the AUEB Natural Language Processing Group for their advice and support.

Summary

The purpose of this is thesis is to investigate several neural network approaches on the task of Aspect-Based Sentiment Analysis. The focus is customer reviews in the English language, concerning the domain of restaurants. We have implemented systems for the tasks of Aspect Category Detection and Sentiment Polarity, as defined in the competition of SemEval 2016. We provide a step by step description of the process followed in the creation of the models and results for the various augmentation techniques, that we incorporated. Our systems are evaluated against the baselines provided by the competition and the best performing systems from previous contestants. At the end we achieve notable results in both tasks, and we make the argument that neural networks can achieve good results even with a limited dataset, if they are combined with the appropriate augmentation techniques.

Περίληψη

Σκοπός αυτής της εργασίας είναι να διερευνήσει μερικές προσεγγίσεις νευρωνικών δικτύων για το πρόβλημα της ανάλυσης συναισθήματος βασισμένης σε όψεις (Aspect-Based Sentiment Analysis). Επικεντρωνόμαστε σε κριτικές πελατών στην αγγλική γλώσσα που αφορούν τον τομέα των εστιατορίων. Έχουμε υλοποιήσει συστήματα για την ανίχνευση της κατηγορίας της όψης (Aspect Category Detection) και του συναισθήματος (Sentiment Polarity), όπως αυτά ορίζονται στον διαγωνισμό του SemEval 2016. Περιγράφουμε βήμα προς βήμα την διαδικασία που ακολουθείται για τη δημιουργία των μοντέλων και παρουσιάζουμε τα αποτελέσματα από τις διάφορες τεχνικές προσαύξησης των δεδομένων, που χρησιμοποιήσαμε. Τα συστήματά μας αξιολογούνται συγκρίνοντας με τα συστήματα αναφοράς (baselines) του διαγωνισμού, καθώς και με τα καλύτερα ως τώρα συστήματα από προηγούμενους διαγωνιζόμενους. Στο τέλος επιτυγχάνουμε αξιοσημείωτα αποτελέσματα και στα δύο προβλήματα και ισχυριζόμαστε ότι τα νευρικά δίκτυα μπορούν να επιτύχουν καλά αποτελέσματα ακόμη και με περιορισμένα δεδομένα εκπαίδευσης, αν συνδυαστούν με τις κατάλληλες τεχνικές προσαύξησης δεδομένων.

Table of Contents

1. Introduction **5**
1.1. Contribution 7
1.2. Notation 8
1.3. Outline 9

2. Aspect Category Detection **10**
2.1. Task Description 10
2.2. Datasets 11
2.3. Related Work 12
2.4. Evaluation Measures and Baselines 13
2.4.1. Evaluation Measures 13
2.4.2. Baselines 14
2.5. Suggested Methods 14
2.5.1. Embeddings 15
2.5.2. Eliminating Unseen Categories 17
2.5.3. Model Creation and Evaluation 18
2.5.4. Data Augmentation Techniques 31
2.5.5. Final Results and Comparisons 34

3. Sentiment Polarity **35**
3.1. Task Description 35
3.2. Related Work 35
3.3. Evaluation Measures and Baselines 36
3.3.1. Evaluation Measures 36
3.3.2. Baselines 37
3.4. Suggested Methods 37
3.4.1. Create and Evaluate Systems 37
3.4.2. Data Augmentation using Neural Machine Translation 49
3.4.3. Final Results and Comparisons 51

4. Conclusions and Future Work **52**
4.1. Conclusions 52
4.2. Future Work 52

References **54**

Appendix **58**

List of Tables and Figures

Tables

Table 1 : Notation table. 9
Table 2 : Dataset for the first Subtask. 13
Table 3 : Evaluation of Augmentation Techniques in ACD. 33
Table 4 : Final evaluation of the suggested model for ACD. 34
Table 5 : Extra instances for the SP task. 35
Table 6 : Data after using Neural Machine Translation in SP. 50
Table 7 : Final Evaluation for the suggested model for SP. 52

Figures

Figure 1.1. First step of ABSA. 6
Figure 1.2. Second step of ABSA. 6
Figure 1.3. The tasks of the competition of SemEval 2016.
Figure 2.1. All the possible categories for Aspect Category Detection. 12
Figure 2.2. The format of the dataset in the SemEval competition. 12
Figure 2.3. A visualization of the word embeddings. 18
Figure 2.4. Removed unseen categories. 19
Figure 2.5. A self attention mechanism to encode the word embeddings (Att model). 21
Figure 2.6. Evaluation of the Att model. 23
Figure 2.7. Evaluation of the Att-LSTM model. 24
Figure 2.8. Evaluation of the G-Att-LSTM model. 26
Figure 2.9. Evaluation of the PR-G-Att-LSTM model. 28
Figure 2.10. The final model (PR-G-Att-BiLSTM). 30
Figure 2.11. Evaluation of the PR-G-Att-BiLSTM model. 30
Figure 3.1. An attention model on top of a concatenation of word and aspect embeddings (Asp-Att model). 40
Figure 3.2. Evaluation of the Asp-Att model. 40
Figure 3.3. Evaluation of the Asp-Att-LSTM model. 42
Figure 3.4. Evaluation of the Dr-Asp-Att-LSTM. 43
Figure 3.5.The Dr-HidAsp-Att-LSTM model. 44
Figure 3.6. Evaluation of the Dr-HidAsp-Att-LSTM model. 45
Figure 3.7. Evaluation of the Pr-Dr-HidAsp-Att-LSTM model. 47
Figure 3.8. Final model (Pr-Dr-HidAsp-Att-BiLSTM). 48
Figure 3.9. Evaluation of the Pr-Dr-HidAsp-Att-BiLSTM. 49
Figure 3.10. Evaluation of the Pr-Dr-HidAsp-Att-BiLSTM after NMT (Neural Machine Translation). 52

1. Introduction

People tend to trust the reviews of other consumers before they make a purchase, but they do not have the time to read all of them, especially if they are lengthy. Thus, it becomes evident that we need a good way to summarize opinions. This summarization also gives a good insight of the consumers' needs, that the companies can make use of in order to provide better services and products. In this thesis, we focus on restaurant reviews in the English language and we try to explore new approaches on the task of Aspect-Based Sentiment Analysis (ABSA). This task is essentially offering a fine-grained summarization of reviews. A typical restaurant or fast food chain is being reviewed on several different entities and the opinion on each of them may be totally different e.g. "*The food was good, but the service was awful* ". In such cases we cannot simply extract a general sentiment from the sentence, because it would be controversial and it would not offer much information. We have to evaluate each entity separately. In most cases, this is implemented by asking the customers to give a rating for each entity that is considered important and taking an average of the scores. An issue that occurs is that we cannot ask the customer to complete a form reviewing all possible entities, because it would be time-consuming and it would ruin the user experience. Furthermore, we have cases where the same entity may have aspects of contradicting polarity "*The food was good, but pricey*". In this case the entity food is considered to have a positive aspect of quality and a negative aspect of price. This makes it even more complex if we request from the user to review each aspect of each entity separately. ABSA comes as a solution to these issues, by being able to take as input unstructured reviews and output the polarity of each attribute for each of the entities. A good example of ABSA would be the one that follows. First by looking to keywords we want to identify the entities and their attributes. Below the entity is separated from the attribute with a "*#"* symbol and together they form an entity#attribute pair (E#A pair).

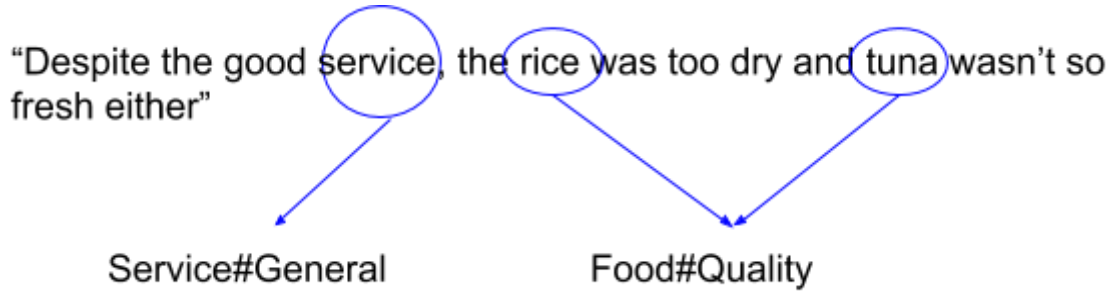


Figure 1.1. First step of ABSA.

The next step is to give a rating to the identified E#A pairs based on words that characterize these pairs.

Figure 1.2. Second step of ABSA.

The well-known competition of SemEval introduced ABSA as a task for the first time in 2014 (M. Pontiki et al., 2014). The following years this task was refined (M. Pontiki et al., 2015), (M. Pontiki et al., 2016) and new well annotated datasets were provided. The year 2016 ABSA ended up having multiple specific subtasks and datasets, that covered 7 domains and were provided in 8 languages . The fact that the competition was offering a fine-grained dataset with well specified tasks, made it an ideal candidate for us in order to explore new techniques on ABSA. In this manner, we would not be confused by the many variations of ABSA, that occur due to a different annotation of the available dataset and specific business needs. Moreover, we would be able to use the systems proposed by previous contestants as our baselines. ABSA was Task 5 in the competition of SemEval 2016 and included several subtasks:

1. The first subtask is named Sentence-level ABSA, meaning that we need to apply ABSA on each sentence. This subtask consists of 3 slots:
    a. In Slot 1 (Aspect Category Detection) we are given a sentence and predefined inventories of E#A pairs and our goal is to identify the E#A pairs that are present in the sentence.
    b. In Slot 2 (Opinion Target Expression) we should identify the linguistic expression used in the given text, that helped us refer the reviewed E#A pair.
    c. In Slot 3 (Sentiment Polarity) we are given which E#A pairs are present in the sentence and we need to identify the polarity towards these pairs.
2. The second subtask is a Text-level approach, where we are given the whole review of each customer and want to find E#A pairs and their polarities, that summarize the review.
3. Finally, in the third subtask, named Out-of-domain ABSA, we can evaluate our systems in a different domain, for which we have not been given any training data.

Since the competition allows participants to select subtasks or even specific slots and create systems only for them, we decided we would focus on the first subtask and implement systems for the slots 1 and 3. The reason behind this decision is that we believe that these slots are the core components of ABSA.

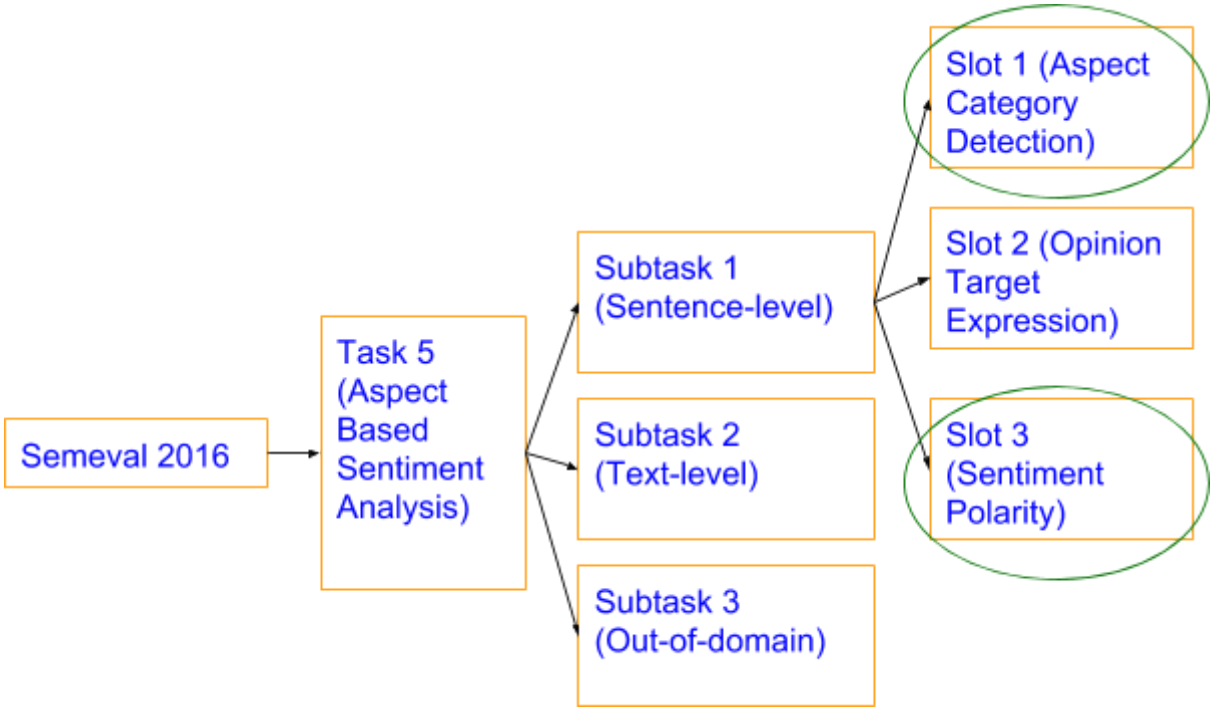


Figure 1.3. The tasks of the competition of SemEval 2016.

1.1. Contribution

In the previous years, the vast majority of the contestants in the competition of SemEval, implemented systems that were based on traditional machine learning techniques. Some top competing teams used Support Vector Machines (SVMs) (Vapnik, 1998) and Conditional Random Fields (CRFs) (Lafferty et al., 2001). Recently, some contestants introduced neural networks to their systems and achieved notable results. The purpose of this thesis is to further investigate a few neural network approaches for this task, since we feel there is margin for improvement. Several of the techniques and systems that we implemented, were initially suggested by previous published work. We decided we would adapt them to the current task and make a few improvements, that we deem necessary. We implemented a system for ACD (Aspect Category Detection) and SP (Sentiment Polarity) and we evaluated against the currently best performing systems. To the best of our knowledge we hold the best result in ACD and one of the top 3 results in SP. In SP we are getting outperformed by models that heavily rely on hand-crafted features. We provide a good documentation of all the approaches that were employed, combined with the source code for any further experimentation. Finally, one of the methods employed

in this thesis for text augmentation, using neural machine translation, was made into a standalone library.

1.2. Notation

The rest of this thesis follows the abbreviations mentioned below :

Abbreviation	Meaning
ABSA	Aspect-Based Sentiment Analysis
SVMs	Support Vector Machines
CRFs	Conditional Random Fields
CNNs	Convolutional Neural Networks
LSTMs	Long Short Term Memory (are a building unit for layers of a recurrent neural network)
ATE	Aspect Term Extraction
HMM	Hidden Markov Model
E#A pair	Entity Attribute pair
ACD	Aspect Category Detection
OTE	Opinion Target Expression
SP	Sentiment Polarity
Tanh	Hyperbolic Tangent Function
ReLU	Rectified Linear Unit Function
SMOTE	Synthetic Minority Oversampling Technique
K-NN	K-nearest neighbours algorithm

Table 1 : Notation table.

1.3. Outline

The remainder of this thesis is organized as follows :

Chapter 2 - Aspect Category Detection: A description of the most popular methods that have been employed so far for aspect category detection, as well as the system that is proposed by this thesis. Our system is an attention-based LSTM, combined with various augmentation techniques.

Chapter 3 - Sentiment Polarity: A description of the most popular methods that have been employed so far for sentiment polarity classification, as well as the system that is proposed by this thesis. Our system is an attention-based LSTM, that uses as input a concatenation of word and aspect embeddings.

Chapter 4 - Conclusions and Future Work : A summary of the results of both systems. A description of the challenges that were faced and ideas for future work.

2. Aspect Category Detection

2.1. Task Description

The first task that we addressed was the ACD (Semeval_2016,Task 5, Subtask 1, Slot 1). In this case, we were given an inventory of predefined entities and attributes, and we had to predict in each sentence, which E#A pairs were present. We have to note that more than one E#A pair may be present in each sentence, or we may not have one. The predefined entities are: RESTAURANT, FOOD, DRINKS, AMBIENCE, SERVICE, LOCATION. The predefined attributes are: GENERAL, PRICES, QUALITY, STYLE_OPTIONS, MISCELLANEOUS. If we combine each entity with each attribute (e.g. RESTAURANT#GENERAL and RESTAURANT#PRICES), we end up with a total of 30 categories, which are all the possible output results.

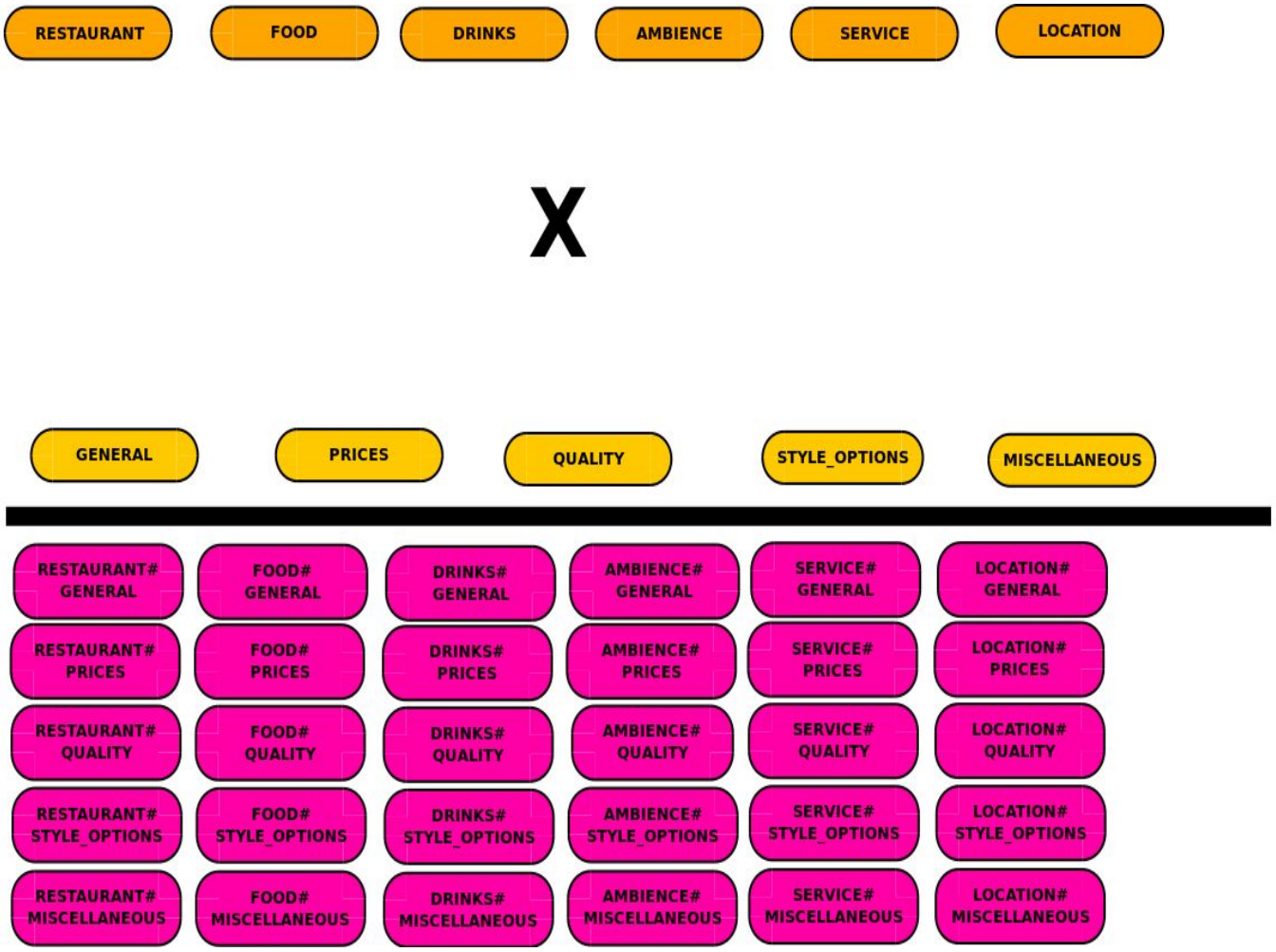


Figure 2.1. All the possible categories for Aspect Category Detection.

2.2. Datasets

The competition of SemEval 2016 (M. Pontiki et al., 2016) offered a total of 39 datasets, from which 19 were for training and 20 for testing. The texts cover 7 domains and were provided in 8 languages. To be more specific, the languages were: English, Arabic, Chinese, Dutch, French, Russian, Spanish, Turkish and the domains: Restaurants, Laptops, Mobile Phones, Digital Cameras, Hotel Museums, Telecommunications. In the Telecommunications domain the data consists of tweets, whereas in the other domains it consists of customer reviews. The texts were manually annotated by 1 or more research groups and several annotation tools were employed. In most cases, the process involved a first step, where multiple people gave the initial annotation and a second step, where contradicting annotations were resolved by an experienced linguist. The annotators were mainly native speakers in order to ensure good results. The datasets with both text-level and sentence-level annotations were provided in an XML format and are available under specific license terms[1]. A good example of the format of the dataset is the one that follows:

```
<sentence id="1004293:3">
<text>The food was lousy - too sweet or too salty and the portions tiny.</text>

<Opinions>
<Opinion target="food" category="FOOD#QUALITY" polarity="negative" from="4" to="8"/>
<Opinion target="portions" category="FOOD#STYLE_OPTIONS" polarity="negative" from="52" to="60"/>

</Opinions>

</sentence>
```

Slot 1 Slot 3 Slot 2

Figure 2.2. The format of the dataset in the SemEval competition.

Above we can see all the sentence-level annotations, that are needed in order to implement all slots in the first subtask. The "*text*" tags include the sentence and just below we have all the opinions. The opinions consist of opinion targets. An opinion target is an expression (word or phrase), towards which a sentiment is expressed. An opinion target belongs to a category (E#A pairs), is placed somewhere in the text and has a polarity (*positive, negative, neutral*). These elements are the expected output of ACD, OTE and SP accordingly. In ACD we have to detect aspect categories, in OTE we

[1] The dataset is available at: SemEval 2016

have to detect the opinion target expressions within the text, and in SP we have, given the previous two to estimate the polarity per expression.

In the domain (Restaurants) and language (English) we are interested in, we have in total 2000 annotated sentences for training and 676 sentences for test.

		TRAIN	TRAIN	TEST	TEST
Language	**Domain**	**#Reviews**	**#Sentences**	**#Reviews**	**#Sentences**
EN	**Restaurants**	**350**	**2000**	**90**	**676**

Table 2 : Dataset for the first Subtask.

2.3. Related Work

The task of ACD (Slot 1) is to identify which aspect and categories are present in each sentence. This task is often complemented by the task of Aspect Term Extraction (ATE) and many researchers propose systems that initially extract the aspect terms and afterwards match them with the proper categories. Since these two topics are so closely related, it is natural that we will mention many papers that are related with the task of ATE. The popularity in the area of ATE both using supervised and unsupervised techniques, has risen in recent years due to their practical applications in ABSA and the contest of SemEval.

Toh and Wang (2014), the winners of SemEval 2014, used supervised methods. Their approach was to extract features similar with those used in traditional name entity recognition systems, by using the training set and word clusters. Toh and Su (2015) suggest using gazetteers and word embeddings for the ATE task and later on for the contest of SemEval 2016 they improved this technique (Toh and Su, 2016) by also using features extracted from a Convolutional Neural Network (CNN). Independently of the feature extraction techniques the problem of ATE is treated as a sequence labeling task. Several of the best teams participating in the SemEval contest have used CRFs and SVMs for sequence labeling (Toh and Wang, 2014), (Toh and Su, 2015), (Chernyshevich, 2014), (Brun, Popa and Roux, 2014), (Brun et al., 2016). Another approach involved a combination of CRFs and HMMs and manually extracted features for sequence classification (Jin and Ho, 2009), (Li et al., 2010). Finally, falling in the previous patterns of supervised learning, are Wang et al. (2016), who proposed applying an RNN to the dependency tree of each sentence for feature extraction and the output is fed to a CRF in order to classify high-level features to labels.

The problem with the supervised methods is that they require a large volume of labeled data for training, so it is no surprise that there are more than a few unsupervised approaches for ATE. Liu et al. (2015) use syntactic rules in order to automatically extract aspects. Another approach was suggested by García-Pablos et al. (2014) and it involved using a graph representation in order to describe the relation between aspect terms and opinion words. He et al. (2017) use an attention mechanism on top of word embeddings, that extracts an aspect embedding, by attempting to emphasize aspect words and deemphasize the irrelevant words. Finally, a recent work by Giannakopoulos et al. (2017) involved a character level LSTM that fed its output to a word level LSTM and on top a CRF in order to label sequences.

2.4. Evaluation Measures and Baselines

2.4.1. Evaluation Measures

The evaluation measure for this slot is the $microF1$ measure, calculated by comparing the annotations that the system returned, with the golden annotations that are provided (using micro averaging). Duplicate E#A pairs in the same sentence were ignored. The $microF1$ is defined as the harmonic mean of the $microPrecision$ and $microRecall$. In order to calculate them we need to first count the $True\ Positives\ (TP)$, $False\ Positives\ (FP)$ and $False\ Negatives\ (FN)$ for each category (recall here that categories are all possible E#A pairs). In case we have n categories the $microPrecision$, $microRecall$ and $microF1$ are given as follows:

$$microPrecision = \frac{TP1+TP2+..TPn}{TP1+TP2+..TPn\ +\ FP1+FP2+...FPn}$$

$$microRecall = \frac{TP1+TP2+..TPn}{TP1+TP2+..TPn\ +\ FN1+FN2+....FNn}$$

$$microF1 = 2 \times \frac{microPrecision \times microRecall}{microPrecision + microRecall}$$

It is evident that this measure is trying to give the incentive to predict as many correct instances as possible (many TP with less FP), while not predicting as positive incorrect instances (many TP with less FN). That means that we need focus on predicting the larger categories correctly. In case we have a class imbalance issue, smaller categories may be ignored, in our attempt to maximize the F1 score. This of course lowers the quality of our predictions.

2.4.2. Baselines

For this slot we will be using 2 baselines. The first was proposed by the competition and the second is the winning system of SemEval 2016 (Toh and Su, 2016) in this slot.

Baseline 1 (SVM): After removing the stop words, an SVM with a linear kernel was trained on 1000 unigram features, extracted from the sentences of each tuple (E#A pair). Similarly, a feature vector is build for each of the test sentences. The trained SVM was given the feature vector of each test sentence and was assigned with the task to give probability score to each E#A pair (e.g. FOOD#QUALITY: 0.6, DRINKS#PRICE: 0.2). A threshold ($t = 0.2$) was responsible for deciding whether the tuple is present in the sentence or not.

Baseline 2 (Single layer feed forward network with features learned from a Deep Convolutional Neural Network): The task is considered to be a multiclass classification problem and is solved by using a set of binary classifiers for each category found in the training data. Essentially, a feed forward network with a single hidden layer is implemented, with the task to identify each time if the category is present in the sentence or not. As input this system was given a combination of hand-crafted features and features extracted from a CNN. In the first case the features that were included are:

a) Word Bigrams (the bigram probability of each word in the training set is used as a feature)
b) Head Words (for each word the head word is extracted from the sentence parse tree and is used as a feature)
c) Name List (the probability of each opinion target occurring in the training data is calculated and a list is formed by keeping only the most frequent opinion targets. Opinion targets may consist of more than one word. We count the word occurrences in the opinion targets, that we previously selected and we form a list by keeping only the most frequent words. By using the later list of aspect words, we calculate the probability for each word in the training set being annotated as an aspect word. The last probability mentioned is essentially our feature)
d) Word Cluster (a number of clusters is created by using the K-means algorithm on word embeddings. The information on which cluster a word belongs to is used as a feature)

Additionally, a CNN was trained using a combination of word embeddings and some of the features, that are mentioned above (Name List, Word Cluster). The CNN outputs a probability for each category. Finally, the authors combine the hand-crafted features with the CNN probabilities and use them as an input to the single layer feed forward network, that will take the final decision.

2.5. Suggested Methods

The structure of this section is briefly described below :

1. Embeddings and train word embeddings on a bigger corpus
2. Eliminating unseen categories
3. Model creation and evaluation

4. Incorporation of augmentation techniques
5. Final Results and comparisons

2.5.1. Embeddings

Since both slots examined in this thesis take as input text, we decided that we would use word embeddings as our primary input to our systems. The data that we had available were not nearly enough in order to train a Word2Vec model (Mikolov et al., 2013), so we decided to use a big dataset in the same domain (restaurant reviews), that was not annotated. That dataset was provided by Yelp as part of the Yelp dataset challenge round 10[2]. Yelp is a company that accumulates customer reviews about restaurants and summarizes them, in order to provide feedback to other potential customers. The dataset contained more than 4.7 million reviews or approximately 40 million sentences with a big diversity, since it involved different regions, businesses and people. We employed the skip gram architecture of Word2Vec. The parameters used for the skip gram are listed below :

- **word_freq > 10** (all words which appear less than 10 times are ignored, since they most probably have some spelling mistake or in general they are not words commonly used)
- **vector_size = 300** (the dimension of the feature vector)
- **iterations = 50** (number of epochs over the corpus)
- **window = 10** (the maximum distance between the current and the predicted word within the window used by Word2Vec)
- **negative_sampling = 10** (number of randomly selected words outside our training window that we try to differentiate from)

A good visualization of the embeddings is important in order to evaluate the quality of the word vectors that we have acquired. The t-SNE (Van Der Maaten and Hinton, 2008) is essentially taking as input a multidimensional vector and plots it in a 2-dimensional space, by trying to lose as little information as possible. The idea is to keep similar similar vectors (in our case word embeddings) close together on the plane, while maximizing the distance between dissimilar vectors. A few examples of the visualization can be seen in the Figure 2.3, where we can see that words with similar meaning appear closer in the plane. On the top image we can see most of the plane and in the bottom image a zoomed in portion of it.

[2]The dataset is available at: Yelp dataset

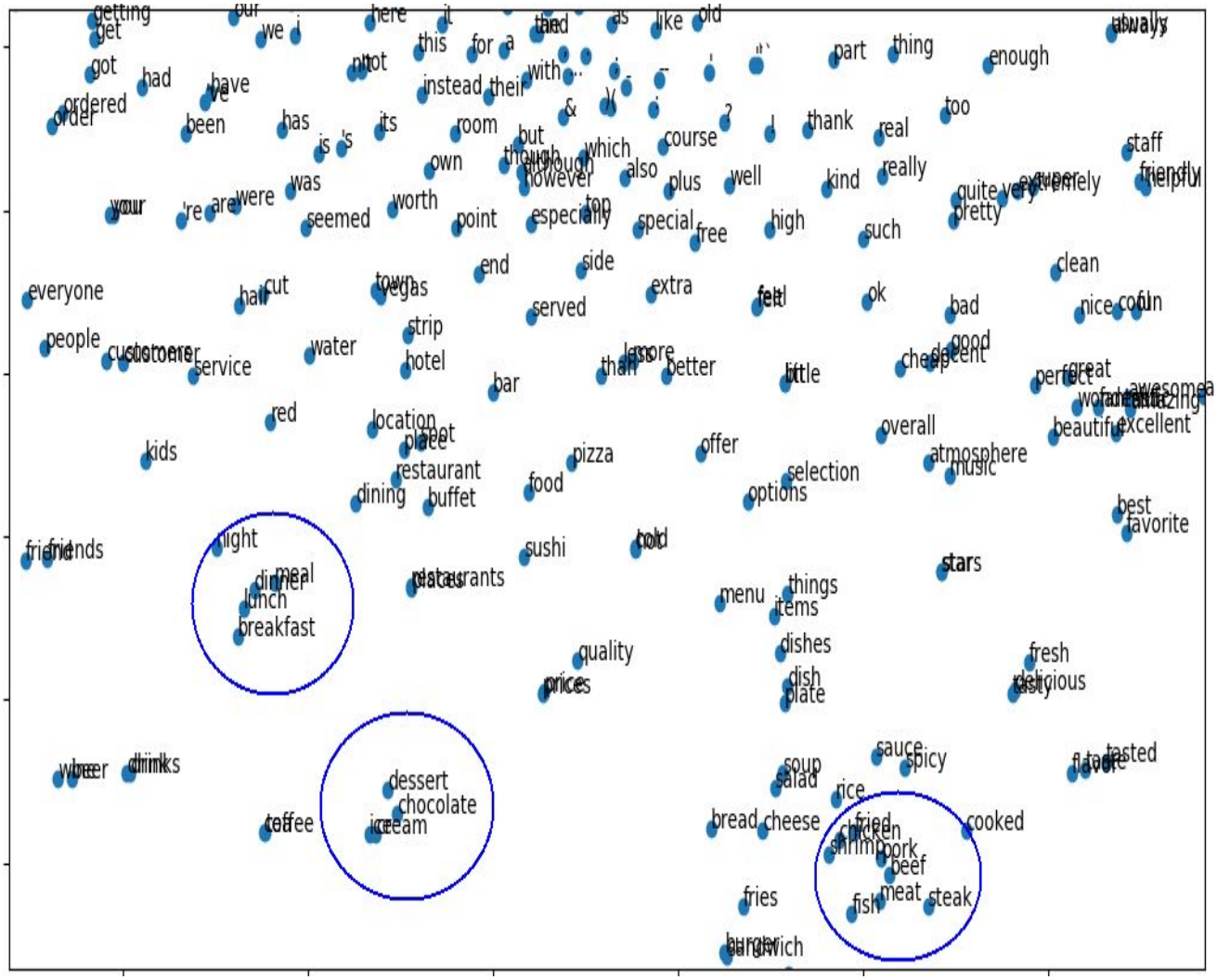


Figure 2.3. A visualization of the word embeddings.

2.5.2. Eliminating Unseen Categories

After having created our primary input (word embeddings), our goal is to identify how we are going to approach the problem. This task can be viewed as a multilabel classification problem, where we have to assign a number of predefined labels-categories to each sentence. The number of possible categories as stated before is 30, which makes the task especially difficult. After some consideration, it was decided that it would be useful to eliminate any possible category, that is not encountered in the training set at least once. The intuition here is that we will not be able to identify a category if we have no instances of it. At the figure below (Figure 2.4), the unseen categories are depicted with gray color. At the end we are left with 12 categories.

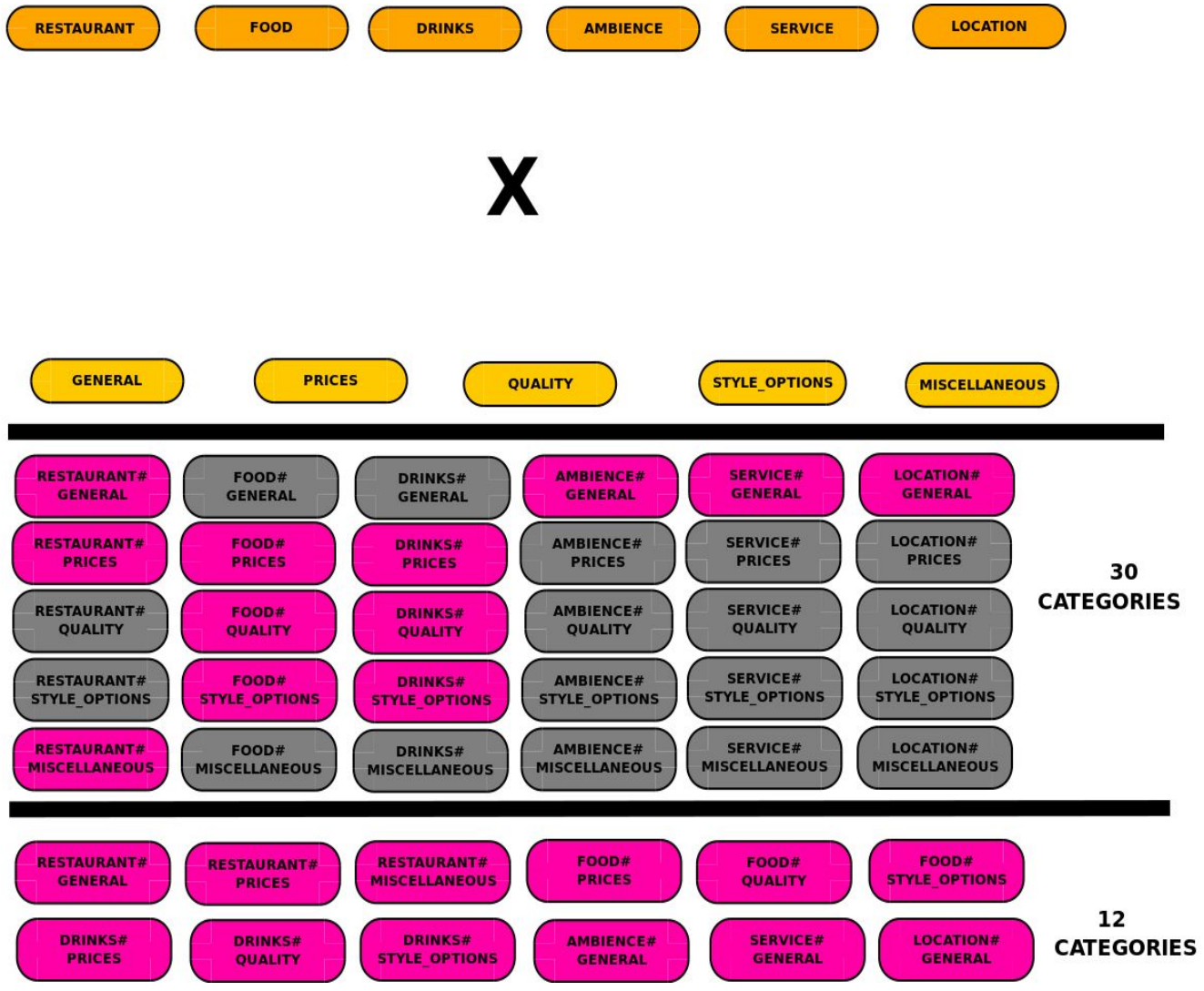


Figure 2.4. Removed unseen categories.

2.5.3. Model Creation and Evaluation

At this point we have established what our input and what the expected output should be. The next step is to define a model that will be trained on the given training dataset and it will produce the output. We will start our experimentation with a simple model and gradually we will be adding components, each time evaluating the performance of the system. In this manner, we will be able to identify how much each new component is contributing to the system. The training data was split for training and validation (70% training, 30% validation). We train the system for a substantial number of epochs and at the end we plot the loss and the micro-F1 at each epoch, in order see how well the model performs on the validation set.

Initial Model:
The initial model that was created takes as input the embedding of each word in the sentence. Directly above the word embeddings we have a simple self attention mechanism, that outputs weights, providing focus to words (Figure 2.5). These weights can then be used to get a weighted combination of the word embeddings and thus we get a representation of the sentence. The discussion below is heavily based on that of Pavlopoulos et al. (2017). More formally, given a vocabulary V, a matrix $E \in R^{m \times |V|}$ containing the m-dimensional vector representations of each word, and a sentence $S = \{w_1, w_2, \ldots, w_k\}$ where $w_1, w_2, ..., w_k$ are the word tokens of S, we can compute the weighted summarization of the word vectors (h_{sum}) in the following manner:

$$h_{sum} = \sum_{t=1}^{k} a_t x_t \quad ,$$

where $x_t \in R^m$ is the vector of the t^{th} word and $a_t \in R$ is the weight proposed from the attention mechanism for that word.

The weights a are proposed by the attention mechanism, that essentially is a Multi-layer Perceptron (MLP) with l number of layers. Below we can see the computations through the layers of the MLP from (1) to l, until the final score (a_t) is produced .

$$a_t^{(1)} = RELU\,(W^{(1)} x_t + b^{(1)})$$

$$\ldots.$$

$$a_t^{(l-1)} = RELU\,(W^{(l-1)} a_t^{(l-2)} + b^{(l-1)})$$

$$a_t^{(l)} = W^{(l)} a_t^{(l-1)} + b^{(l)}$$

$$a_t = softmax(a_t^{(l)}; a_1^{(l)}, ..., a_k^{(l)}),$$

where
$a_t^{(1)}, ..., a_t^{(l-1)} \in R^r$, $a_t^{(l)}, a_t \in R$, $W^{(1)} \in R^{r x m}$, $W^{(2)}, ..., W^{(l-1)} \in R^{r x r}$, $W^{(l)} \in R^{1 x r}$, $b^{(1)}, ..., b^{(l-1)} \in R^r$, $b^{(l)} \in R$.

At the next step, by taking the weighted representation of vectors and adding a sigmoid function, we produce a score (P_{at}) for each category. This score indicates how probable is for an E#A pair to be in the sentence .

$P_{at} = \sigma(W_p h_{sum} + b_p)$, where $W_p \in R^{1 x m}$, $b_p \in R$ and σ denotes the sigmoid function.

Since this is a multilabel classification task, a different sigmoid was used for each category and a probability for each possible E#A pair being in the sentence is calculated. Finally, in order to be able to make a classification we use a threshold of 0.5. The values that are higher or equal to 0.5 are set to 1 and the ones that are lower are set to 0. The value 1 is indicating that the aspect category is present, whereas a value of 0 that it is not. A downside of this model is that the attention only considers information about the respective word and not about its context. We call the system described here Att model.

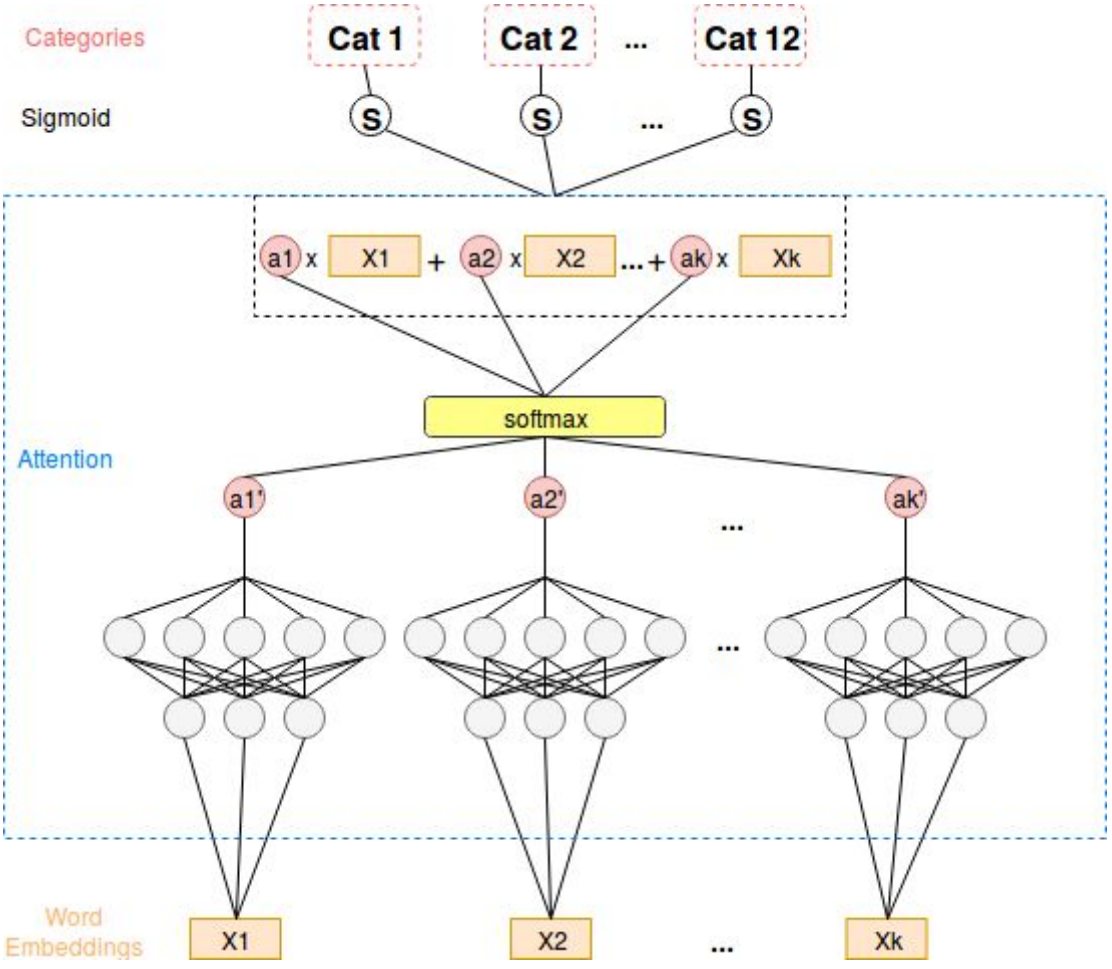


Figure 2.5. A self attention mechanism to encode the word embeddings (Att model).

At Figure 2.6 we can see the performance of the first model after 1000 epochs. The blue line depicts the performance on the training data, whereas the orange line depicts the performance on the validation data. As we can see around the 400th epoch the validation loss no longer falls and the validation F1 no longer increases, whereas the training loss and F1 continue to improve, because of over-fitting. That would be the optimum point to stop our training, and we would get a F1 score of 59%. In a similar manner we evaluate each new addition to our system.

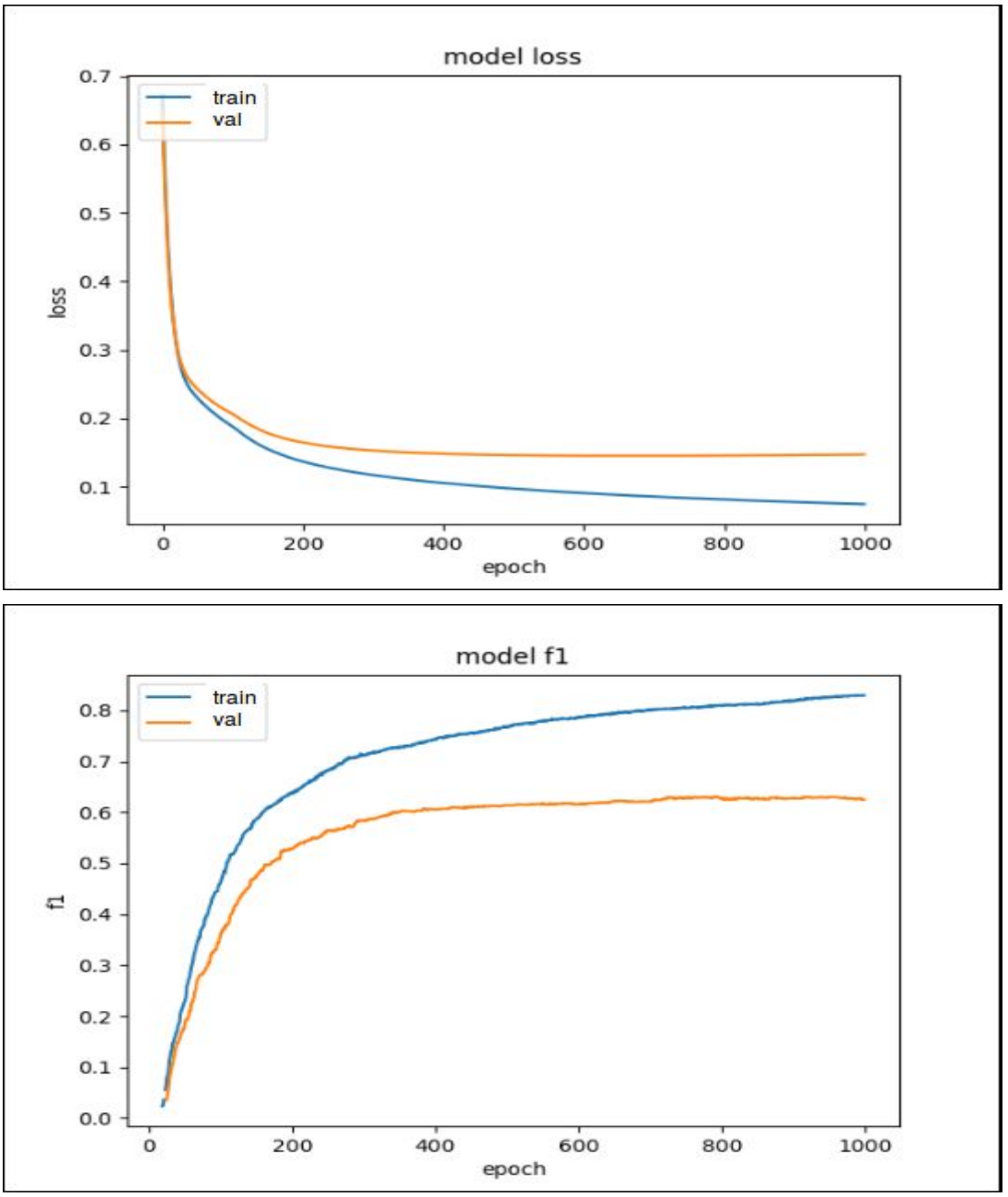


Figure 2.6. Evaluation of the Att model.

Improvement 1:
First, LSTM cells (S. Hochreiter and J. Schmidhuber, 1997) were employed and each hidden state of the LSTM was passed through a self attention mechanism. The discussion below is heavily based on that of Pavlopoulos et al. (2017). Given a set of vectors $x_1, x_2 ..., x_t$, where $x_k \in R^m$ the LSTM produces for each one of them a hidden states $h_1, h_2 ..., h_t$ in the following manner:

$$c_t = f_t \odot c_{t-1} + i_t \odot \overline{c}_t$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$

$$\overline{c}_t = tanh(W_c x_t + U_c h_{t-1} + b_c)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$

$$h_t = o_t \odot tanh(c_t)$$

Where i_t is the input gate that defines how much of the newly computed state for the current input is taken into account; f_t is the forget gate, that defines how much of the previous state is taken into account; o_t is the output gate that defines how much of the internal state is exposed to the external network; $\overline{c}_t$ is the proposed hidden state at position t that takes into consideration the word vector x_t and the previous hidden state h_{t-1} (the initial hidden state h_0 is initialized with zeros); c_t is the internal memory of the unit, that is a combination of the previous memory, multiplied by the forget gate, and the newly computed hidden state, multiplied by the input gate. Intuitively it is a combination of how we want to combine previous memory and the new input; $\odot$ denotes the element-wise multiplication; $W_f, W_i, W_c, W_o \in R^{hxm}$; $U_f, U_i, U_c, U_o \in R^{hxh}$; $b_f, b_i, b_c, b_o \in R^h$.

The self attention mechanism stays unchanged but now takes as input the hidden states that the LSTM produces instead of taking word vectors. The new summarization of the sentence is calculated by using the the hidden states h_t and the weights of the attention mechanism a_t.

$$h_{sum} = \sum_{t=1}^{k} a_t h_t$$

The probability computation stays unchanged. The LSTM cells are good at encoding sequences, so hopefully we will get a better representation of the sentence than the previous model. We get a better representation because the attention mechanism now receives a more informative input, which takes into consideration not only the word vector, but also information about preceding words. We call this model Att-LSTM. By looking at Figure 2.7 the optimum epoch to stop training would be around the 50th epoch, where we would get a validation F1 score of 63%. We get a reasonable improvement in

F1, but just after the 50th epoch we can see a sudden increase in the loss, which is most probably indicating that the model is overfitting.

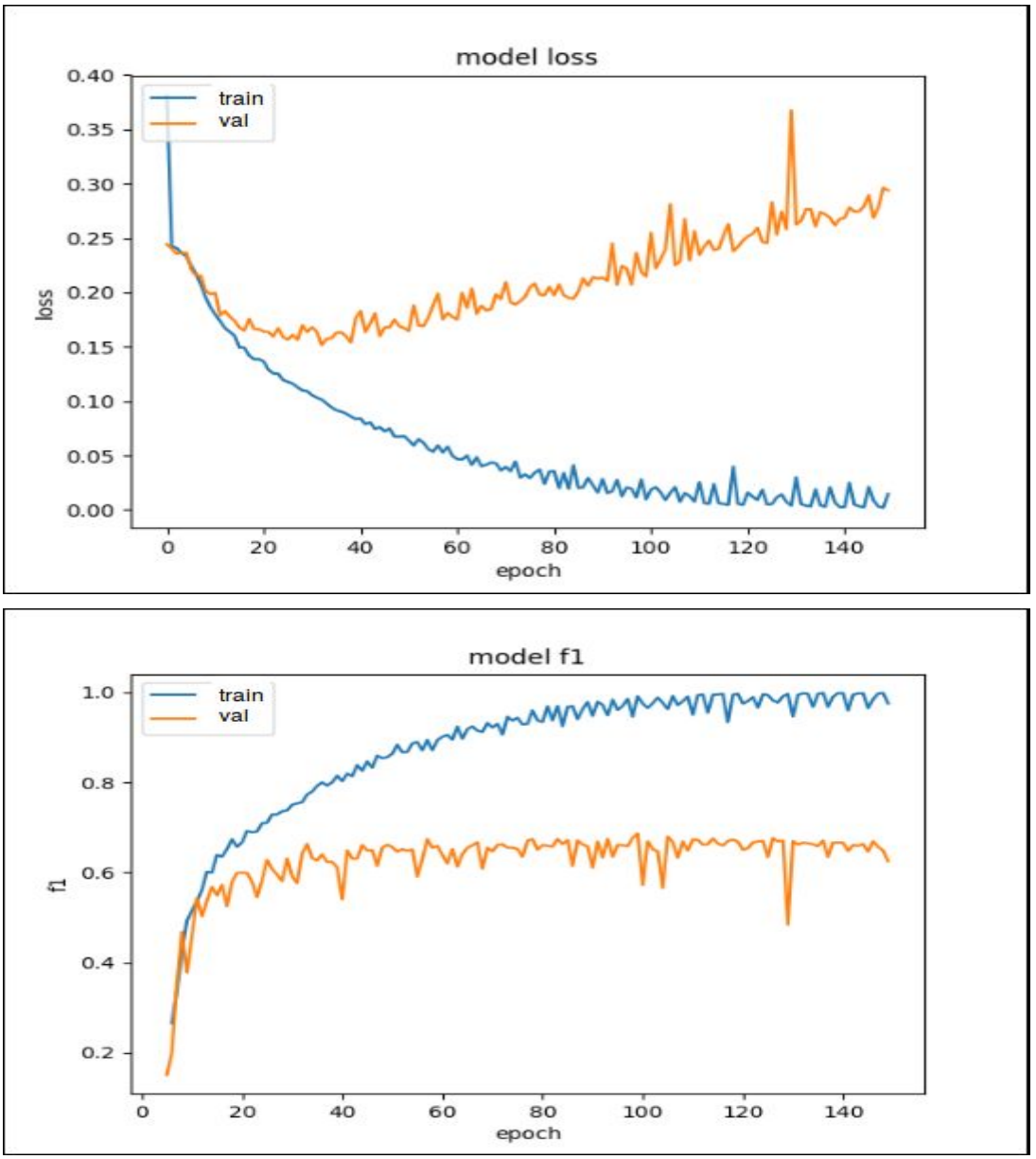


Figure 2.7. Evaluation of the Att-LSTM model.

Improvement 2:

In order to tackle the problem of overfitting we decided to add Gaussian noise to the embeddings and two dropout layers (Srivastava, 2014). By adding Gaussian noise we mean that at each epoch, for each word embedding a new noise vector is generated, by drawing from a Gaussian distribution, with mean equal to zero and standard deviation equal to 0.5. This noise vector is created with a probability that we define as a hyperparameter. Given a word vector $x_t \in R^m$ and a noise vector $G_t \in R^m$, we

denote the i_{th} dimension of each one of them as $x_t^i \in R$ and $G_t^i \in R$ accordingly. Each dimension of a word vector will be transformed in the following manner:

$$\overline{X}_t^i = x_t^i + G_t^i$$

$$G_t^i \sim N(0, 0.5)$$

This is nowadays a well-known domain agnostic augmentation technique, that helps the word embeddings generalize better (T. Devries and G. W. Taylor, 2017). As for the dropout, one layer was inserted just after the embeddings and one after the attention layer. The dropout layers help us, by randomly dropping a few dimensions of the previous layers, so our decision will not be strongly based on specific dimensions. Moreover, dropout is helpful for data augmentation since the same input with different dimensions dropped each time is considered as a new instance in our network. At each sentence the dropout mask is kept the same for all the words. This means that at each sentence we drop the same dimensions from the word embeddings. We call this model G-Att-LSTM. By taking a look at the loss at Figure 2.8 we can see that there are fewer spikes. We have managed to tackle the issue of overfitting to some extent and we have an even better result (at the 90th epoch we get a F1 score of 68%).

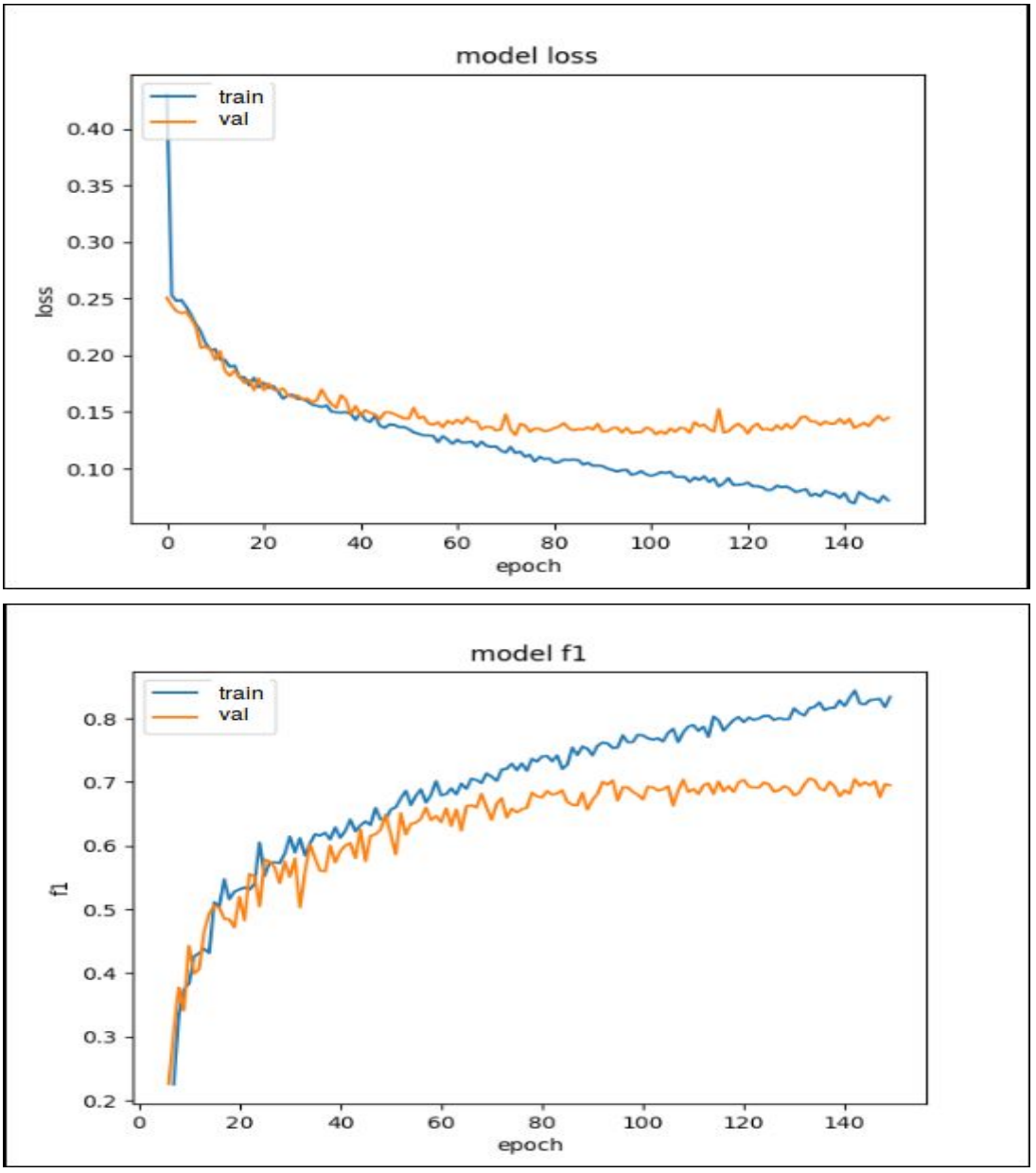


Figure 2.8. Evaluation of the G-Att-LSTM model.

Improvement 3:

Next, we added projection layers after the embeddings and before our output layer. The projection layers help our model learn only the most useful combination of dimensions of the previous layer. After some experimentation on the development data the activation function of the first projection layer was decided to be *tanh* and of the second one ReLU. Given sentence with k words $S = \{w_1, w_2, \ldots, w_k\}$ and accordingly their vector representations $\{x_1, x_2, \ldots, x_k\}$, where $x_t \in R^m$ we can define a projection layer that gives the following output for each word vector:

$J = tanh\,(W_j x_t + b_j)$, where $W_j \in R^{s\,x\,m}$, $b_j \in R^s$ and $s \ll m$

We use this layer in order to project each word vector from m to s dimensions and at the end we concatenate all the projections of the word vectors into a single flattened vector of size ($s \cdot k$). We have to keep in mind that the same projection matrix is used for all the words of the sentence. The reason s is much smaller than m is that we need to select only the most useful combinations of the dimensions of the word vectors. In a similar manner works the other projection layer with the difference that it uses the activation function ReLU. We call this model PR-G-Att-LSTM. By looking at Figure 2.9 we see that the optimal epoch to stop training is around the 70th epoch, where we achieve a F1 score of 71%.

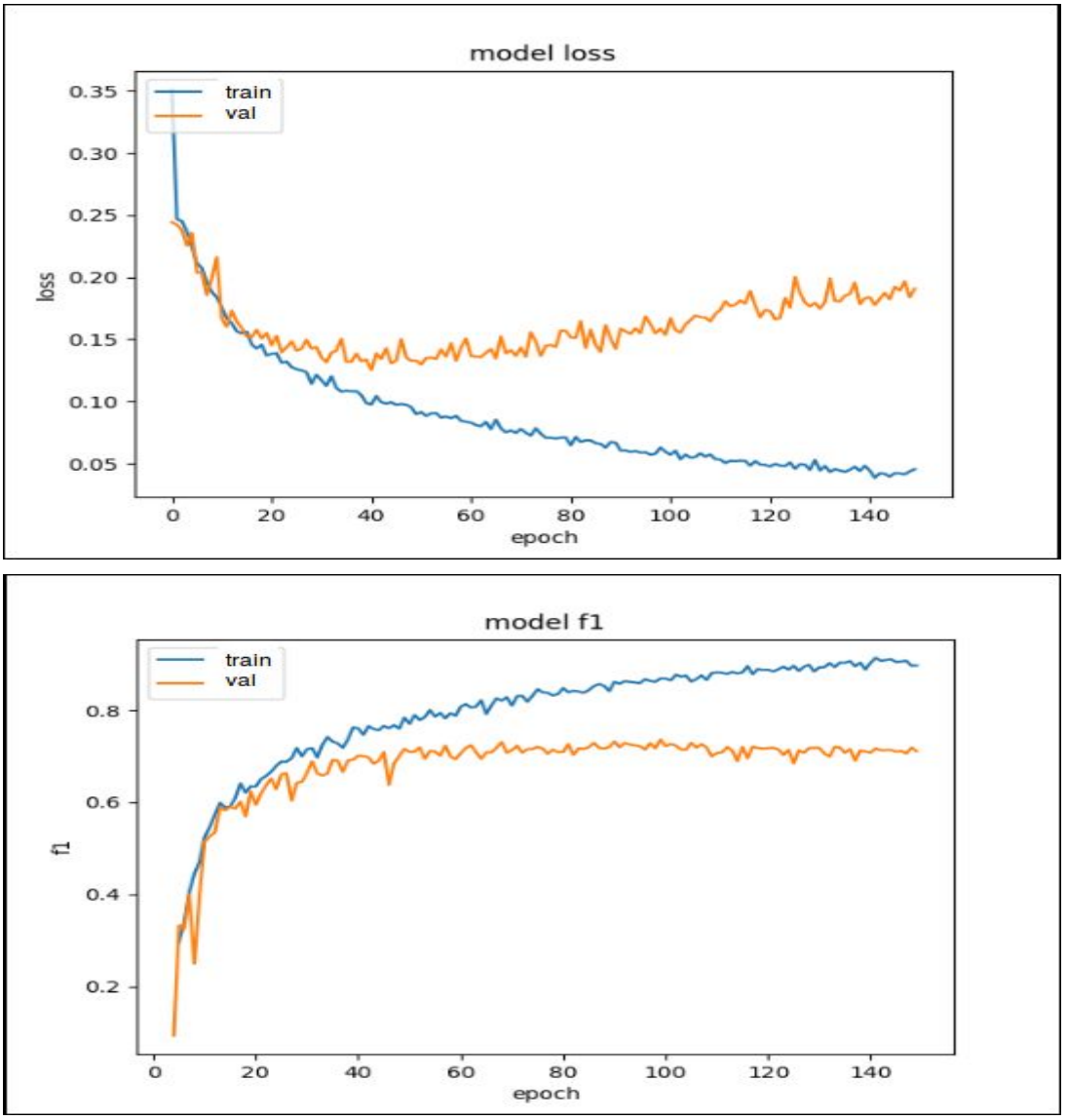


Figure 2.9. Evaluation of the PR-G-Att-LSTM model.

Final model :

Finally, we make our LSTM bidirectional, meaning that we will use one LSTM cell to encode the sentence from left to right and another one to encode from right to left. We will then concatenate the output and hidden states of each cell (Figure 2.10). Given the hidden state produced by the LSTM that

goes from left to right for the t^{th} word ($h_t^{\rightarrow}$) and the hidden state of the LSTM that goes from right to left ($h_t^{\leftarrow}$) the final hidden state (h_t) that the Bi-LSTM will produce is computed as follows:

$$h_t = (h_t^{\rightarrow}; h_t^{\leftarrow})$$

We simply concatenate the hidden state at each timestep. This is nowadays a typical technique that boosts performance. The intuition behind this alteration is that now you provide an even better input to the attention mechanism, by taking into consideration words both before and after the given vector. We call this model PR-G-Att-BiLSTM. Another step that we need to take, is fine-tuning our model. This is done by selecting a few possible values for each parameter we want to evaluate and running our model multiple times, each time with different settings. At the end we keep the settings with the best performance on the validation set. By looking at Figure 2.11 we can see that the optimal stop for our training would be the 60th epoch, and we would get a F1 score of 73%.The optimum hyper-parameters for the PR-G-Att-BiLSTM model are available at Appendix-Table A.

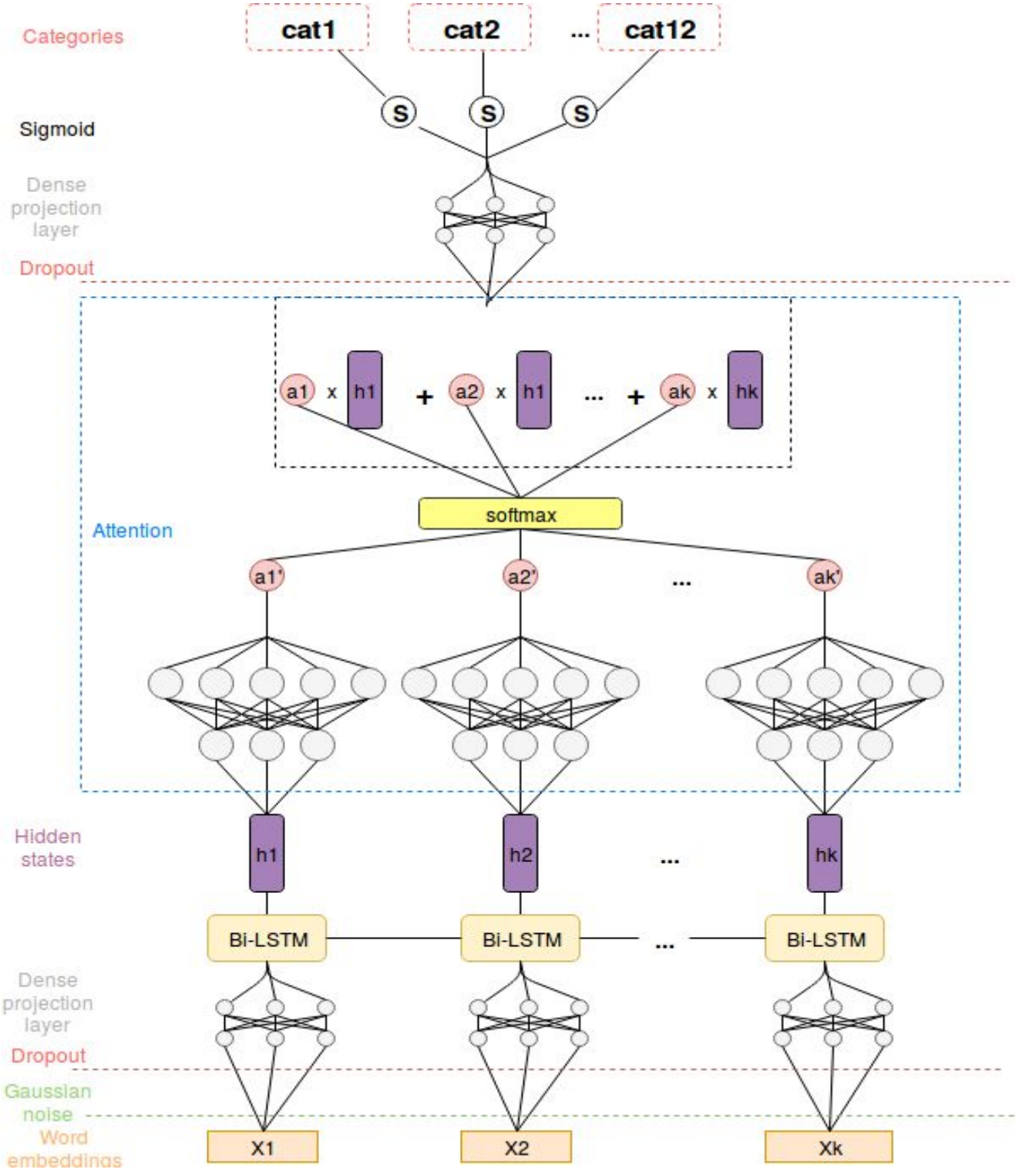


Figure 2.10. The final model (PR-G-Att-BiLSTM).

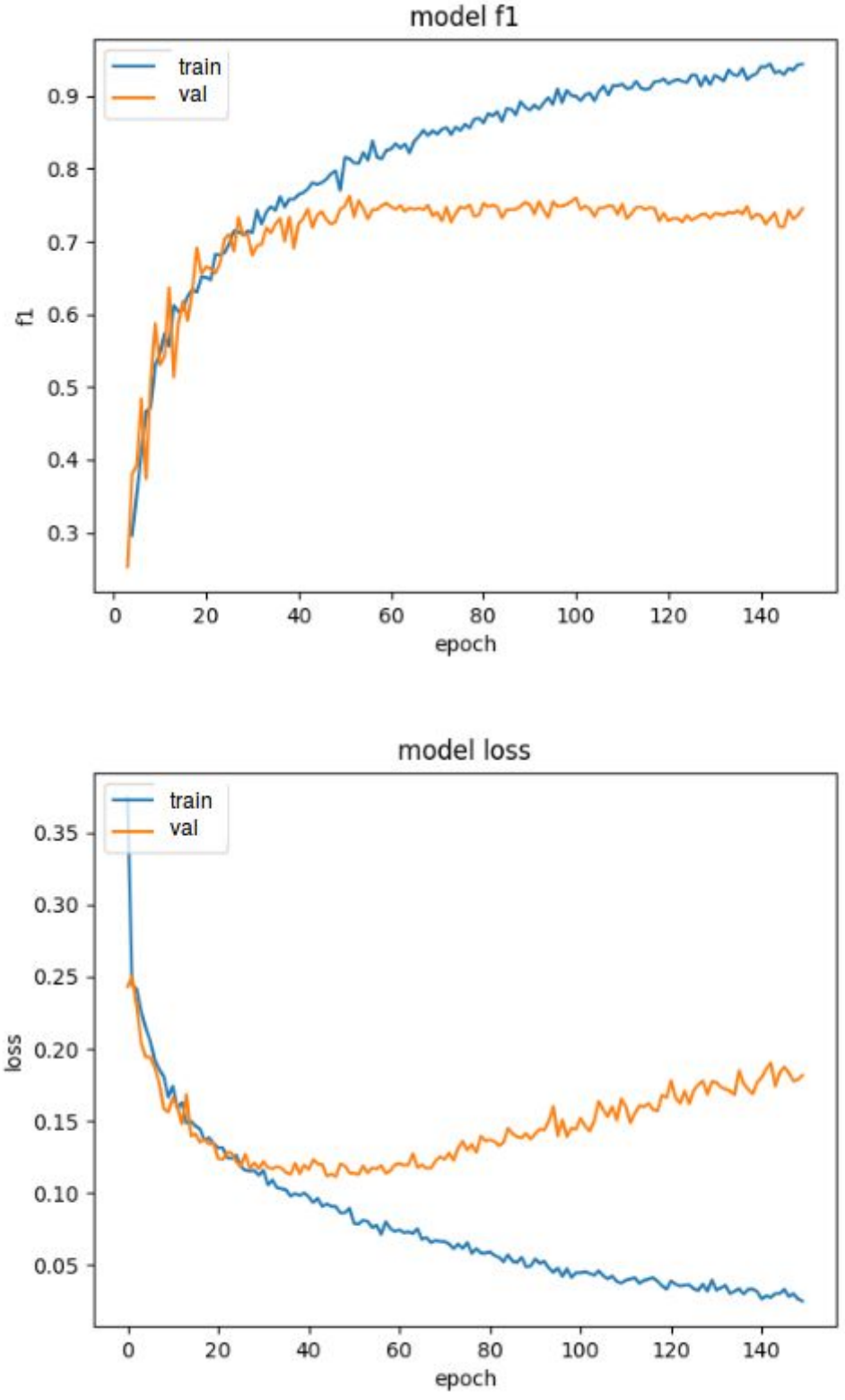


Figure 2.11. Evaluation of the PR-G-Att-BiLSTM model.

2.5.4. Data Augmentation Techniques

One of the main issues of ACD was the class imbalance. There were some classes that were represented by only a handful of instances and others by a few hundreds. Initially both undersampling and oversampling techniques were considered to handle this problem, but since we had a limited amount of data, the oversampling was deemed more useful. The techniques that we experiment with here are the Synthetic Minority Oversampling Technique or in short SMOTE (N. Chawla and K. Bowyer, 2002) and augmentation via neural machine translation. Both techniques do data augmentation, by creating new instances. Neural networks benefit from large datasets, so we can get a better performance by increasing the training data. SMOTE is nowadays considered one of the standard methods for handling imbalanced classes. The general idea behind it is to create new synthetic samples of a category by using the K-NN algorithm. Given a word vector SMOTE finds it's closest neighbours (Euclidean distance) and produces a new vector by averaging the neighbors found. By doing this for every word in the sentence we get a new neighbouring sentence. As for the second technique, it relies on the idea that when we translate text in one language and then retranslate it back to the original language, the text will have changed slightly, but hopefully it will have contained the same meaning. The languages that were considered to be more appropriate are: English, Spanish, German, French. Between all of these languages we get high quality translations. A few examples of the procedure are listed below:
Original sentence: "*That was the best meal I have had in ages*"

a. Translate from English to German and back to English: "*This was the best meal I have had in years.*"
b. Translate from English to French and back to English: "*It was the best meal I had in a long time.*"

Table 3 below describes the results of applying the augmentation techniques.Columns 2-3 of the table are:

- **Train** column, where we can see all the instances we initially had in each category.
- **SMOTE** column shows how many new instances were created with this technique.

The rest of the columns depict the instances created by neural machine translation. The codes of the languages in use are: EN = English, DE = German, ES = Spanish, FR = French. In order to describe that a sentence is translated from one language to another we will use the origin language code followed by an arrow ($->$) and finally the target language code. A few examples are: $EN -> DE$ means we translate from English to German, $EN -> DE -> EN$ means we translate from English to German and back to English. The remaining columns of Table 3 are:

- **Trans(allx2)** column, where all instances are doubled ($EN -> DE -> EN$). We create a new instance, by taking the original sentence translating it from English to German and back in English. The idea here is that we just want to create more data.
- **Trans(minor x3 single)** column, where the instances of the major categories are doubled $(EN -> DE -> EN)$, whereas the instances of the smaller ones are tripled $(EN -> DE -> EN)$ and $(EN -> FR -> EN)$. Here we use single translation meaning we use only one middle language in order to create a new instance. The idea here is that we want

to make the imbalance between the categories as small as possible, so we create more instances of the small categories.

- **Trans(minor x3 double)** column, where the instances of the major categories are doubled $(EN -> DE -> EN)$, whereas the instances of the smaller ones are tripled $(EN -> DE -> EN)$ and $(EN -> DE -> FR -> EN)$. When we want to get two new instances of the same sentence we can get one instance by translating with a single middle language and another instance by translating with two (double) middle languages. The idea here is that if we want to create two new instances of the same sentence the second one should be a bit different.
- **Trans(major x 2)** column, where only the major categories are doubled $(EN -> DE -> EN)$ and the smaller ones are left the same. The idea is that the small categories just add more noise to our system since only a few instances of them will be encountered during the evaluation, so we just need to augment the bigger categories in order to achieve good results.

At the bottom of the table we have the evaluation of each technique on the validation data, by using the PR-G-Att-BiLSTM model that we proposed. As we can see the best performing technique is using neural machine translation and simply doubling all the instances **Trans(allx2)**. The **Trans(minor x3 single)** and the **Trans(minor x3 double)** seem to help to some extent as well, but by generating more than two instances of the same sentence we are most probably replicating it rather than creating a similar one. In the **Trans(major x 2)** we get a bad performance because by augmenting only the major classes we essentially increase the imbalance. The worst results were produced by SMOTE probably due to the fact that it could not create a good quality synthetic sentence. By simply selecting neighbouring words we may change the meaning of the sentence by a lot. This is most probably introducing noise to our training data.

Categories	**Train**	**SMOTE**	**Trans(allx2)**	**Trans(minor x3 single)**	**Trans(minor x3 double)**	**Trans(major x2)**
Restaurant # General	421	421	**842**	**842**	**842**	**842**
Restaurant # Prices	80	**180**	**160**	**240**	**240**	**160**
Restaurant # Miscellaneous	97	**197**	**194**	**291**	**291**	**194**
Food # Prices	82	**182**	**164**	**246**	**246**	**164**
Food # Quality	681	681	**1362**	**1362**	**1362**	**1362**
Food # Style_Options	128	128	**256**	**384**	**384**	**256**
Drinks # Prices	20	**120**	**40**	**60**	**60**	20
Drinks # Quality	46	**146**	**92**	**138**	**138**	**92**
Drinks # Style_Options	30	**130**	**60**	**90**	**90**	30
Ambience # General	226	226	**452**	**452**	**452**	**452**
Service # General	419	419	**838**	**838**	**838**	**838**
Locations # General	28	**128**	**56**	**84**	**84**	28
	F1 = 73	**F1 = 70.11**	**F1 = 74.20**	**F1 = 73.33**	**F1 = 73.80**	**F1 = 71.20**

Table 3 : Evaluation of Augmentation Techniques in ACD.

2.5.5. Final Results and Comparisons

After concluding that the best model is PR-G-Att-BiLSTM augmented with the Trans(all*x2) technique, the final step is to evaluate the system with Micro-F1, the official measure of SemEval '16 Slot 1. We evaluate against 2 systems. The 1st one is the baseline system proposed by the competition (SVM) and the second one is the winning system of SemEval 2016 (Toh and Su, 2016)(Single layer feed forward network with features learned from a Deep Convolutional Neural Network). To evaluate our model we take the training data, we augment them, we use all of them in order to train our model and we evaluate on the test data, that are provided by the competition. The last model is the model that we are proposing and it outperforms the current best result.

System	**F1 score**
Sem.2016 Slot 1 - Winner	73.03
SVM (Baseline model)	59.92
PR-G-Att-BiLSTM	**74.20**

Table 4 : Final evaluation of the suggested model for ACD.

3. Sentiment Polarity

3.1. Task Description

Given the E#A pairs and the opinion target expressions, that are present in each sentence, we should output the polarity expressed towards these pairs (Semeval2016,Task 5,Subtask 1, Slot 3). The polarity is expected to be in a 3-way scale (*positive, negative, neutral*). Despite using the same dataset as in the previous slot, we have more training instances. That is because, if a sentence has more than one aspect, we consider it to be a new instance for each aspect that is present. The total number of train and test sentences are indicated below.

		TRAIN	TRAIN	TEST	TEST
Language	**Domain**	**#Reviews**	**#Sentences**	**#Reviews**	**#Sentences**
EN	**Restaurants**	**350**	**2507**	**90**	**859**

Table 5 : Extra instances for the SP task.

3.2. Related Work

In the past few years there have been some attempts to create models for aspect-based sentiment analysis, as well as for target-based sentiment analysis. The difference between the two is that in target-based you are given the target, which is a word or a sequence of words, and one needs to find the polarity expressed towards them. Whereas, in aspect-based we are given aspect categories, and we need to indicate the polarity expressed towards them. Both fields have more than a few points in common and models proposed for the one can be applied to the other with minor modifications.

For target-based sentiment analysis we start with the popular method, proposed by Tang et al. (2015). In their system they introduce target information as a feature before encoding the sentence with LSTM cells. Another suggestion was made by Vo and Zhang (2015), who attempt to capture the interaction between the target and the sentiment words. By using multiple embeddings, multiple pooling functions and sentiment lexicons, they manage to extract useful features from tweets in an unsupervised manner. In their approach Tang et al. (2016) use multiple hops in order to create a memory like network. Each hop combines the output of an attention mechanism, that encodes the sentence and the linear representation of the output of the previous hop. Zhang et al. (2016) use a GRNN (D. F. Specht, 1990) to produce the best features on the right and the left of the target word

and pooling functions on top of the hidden states in order to produce the final output. Dong et al. (2014) use a neural network, that based on the context and the syntactic structure, learns whether it should incorporate the sentiment on a specific target word or not. Finally, Liu and Zhang (2017) use attention in order to select the best features on the left and right of the target word. These best context features help them decide the sentiment towards the target word.

As for the aspect-based sentiment analysis we have Ruder et al. (2016) who suggest a hierarchical system, where a Bi-LSTM is used to encode the sentence and this encoding is concatenated with an aspect embedding and passed to another review level Bi-LSTM as input, in order to produce a probability distribution over the sentiments. Ma et al. (2017) proposed an interactive attention model, that uses information from the target in order to learn the context and vise versa. Wang et al. (2016) proposed a concatenation of aspect embeddings and word embeddings as input to an attention-based model, that has the ability, given a different aspect embedding to focus on different parts of the same sentence. Chen et al. (2017) incorporate the distance of each other word from the target in their proposed attention mechanism. The winners of SemEval 2016 in slot 3, Brun et al. (2016), suggest a Feedbacked Ensemble Modeling Process,where a one-vs-all logistic regression model based on linguistic features outputs a model representation and a cross validation score, that are later on used to improve the feature space.

3.3. Evaluation Measures and Baselines

3.3.1. Evaluation Measures

The oficial evaluation measure for this system was accuracy, that is defined as the number of correctly predicted polarity labels of the gold aspect categories, divided by the total number of the gold aspect categories.

$$Accuracy = \frac{Number\ of\ correctly\ predicted\ labels}{Total\ number\ of\ gold\ aspect\ categories}$$

3.3.2. Baselines

Again in this slot we have two baseline systems. The first one was proposed by the competition and the second one was the system implemented by the winner of this slot in 2016 (Brun et al., 2016).

Baseline 1 (SVM):
After removing the stop words, an SVM with a linear kernel was trained on 1000 unigram features extracted from the sentences containing each E#A pair of the training data. In addition, an integer-valued feature, that indicates the category of the pair, is used. The correct label for the extracted training feature vector is the corresponding polarity value (e.g., "*positive*"). In a similar manner a vector is created and classified for each test sentence.

Baseline 2 (Feedbacked Ensemble Modelling):
The initial step is to use the XIP syntactic parser (Brun et al., 2016), combined with a semantic extraction component, in order to extract linguistic features for each term in the sentence. Moreover, the aspect category that is present in the sentence is added as a feature. Each term is replaced by it's aspect category (e.g. sushi -- > FOOD, staff→ SERVICE). In order to make the classification, a one-versus-all Elastic Net regression model (Zou and Hastie, 2005) is used. Perhaps the most important aspect of this implementation is the Feedbacked Ensemble Modeling Process, where the one-versus-all Elastic Net regression model is used to produce a model representation and a cross validation score. Then the model representation and the validation score will be used in order to improve the feature space. This process is repeated several times.

3.4. Suggested Methods

The methodology followed in this section is:

1. Create and Evaluate systems
2. Incorporate proper augmentation techniques
3. Final results and comparisons

3.4.1. Create and Evaluate Systems

In this slot we will be using the same embeddings as in the previous slot. We will be creating models starting from simple ones and gradually adding new components in order to see how their addition affects the performance. The training data is split into 70% for training and 30% for validation. The fundamental functionality of the final model proposed in this chapter is suggested by Wang et al. (2016).

Initial Model:

In SP we have to output the polarity expresed towards the E#A pairs. We have to note that an instance is a sentence with only one aspect (E#A pair) considered. That is because, each sentence has been cloned as many times as the aspects inside. Since in this task we are aware of the E#A pair that is present in the given sentence, we wanted to incorporate this knowledge, as feature to our model.This is implemented by concatenating the embedding of each word in the sentence with the target aspect embedding (Figure 3.1). This of course means, that a specific aspect embedding is repeated as many times as the words of the sentence. On top, we have a self attention mechanism, that outputs the probability of each class (*positive, negative, neutral*), by using a softmax function. The discussion below is heavily based on that of Pavlopoulos et al. (2017). Given a sentence with k vectors $S = \{x_1, x_2, \ldots, x_k\}$,where $x_1, x_2, \ldots, x_k \in R^m$, and the aspect embedding ($A \in R^d$) that is present in the sentence, we can produce the weighted summarization of the sentence in the following manner.

$$h_{sum} = \sum_{t=1}^{k} a_t \bar{x}_t$$

$\bar{x}_t = (x_t; A)$, we simply concatenate the word vector with the aspect vector.

where a_t is the weight proposed from the attention mechanism for that particular word concatenated with the aspect vector ($\bar{x}_t \in R^{d+m}$). The attention score is computed in a similar manner as in ACD (attention weights) with the only difference being that the input is the concatenation of aspect and word vector ($\bar{x}_t$) instead of only the word vector (x_t). At the next step, by taking the weighted representation of vectors and adding a softmax function, we produce a score for each category (the scores sum to 1).

$$P_{(s/a)} = softmax(W_p h_{sum} + b_p), \quad \text{where } W_p \in R^{3 x (d+m)} \text{ and } b_p \in R^3.$$

The $P_{(s/a)} \in R^3$ contains three probabilities ($P_{(pos/a)}, P_{(neg/a)}, P_{(neu/a)}$) . Each of them indicates how probable is for the given aspect to have a specific polarity (*positive, negative, neutra*l). At the end since only one polarity is wanted, the class with the highest score is selected. We call this model Asp-Att. The intuition behind this model is that given the same sentence, but with a different aspect embedding, the attention mechanism will focus on different words each time. The word embeddings used here are pretrained, whereas the aspect embeddings are initialized randomly and learned through backpropagation. By taking a look at Figure 3.2 we can see that the optimal point to stop training our model would be on the 200th epoch, and we would get an accuracy of 78%, whereas the loss seems to drop close to 0.55.

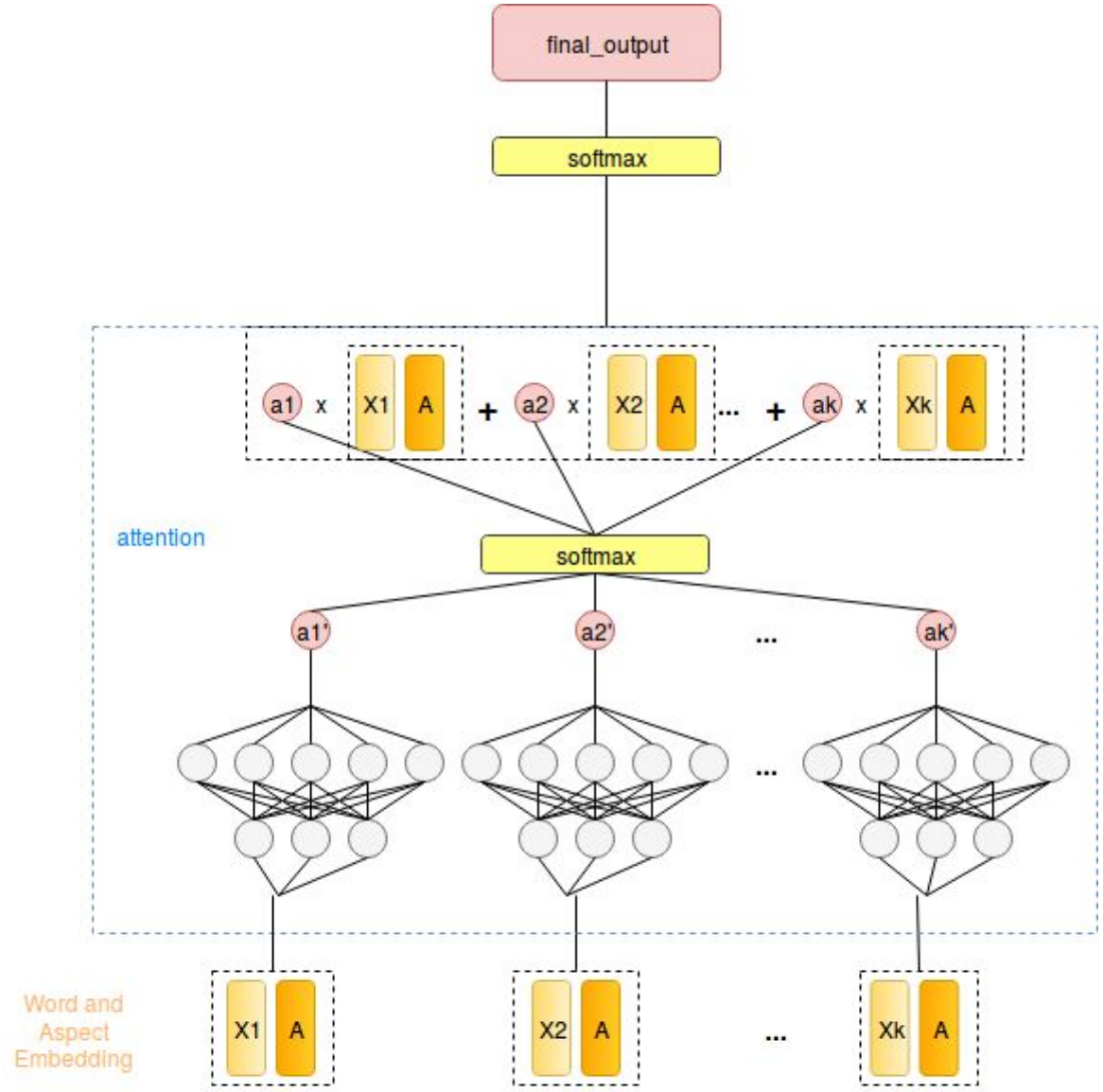
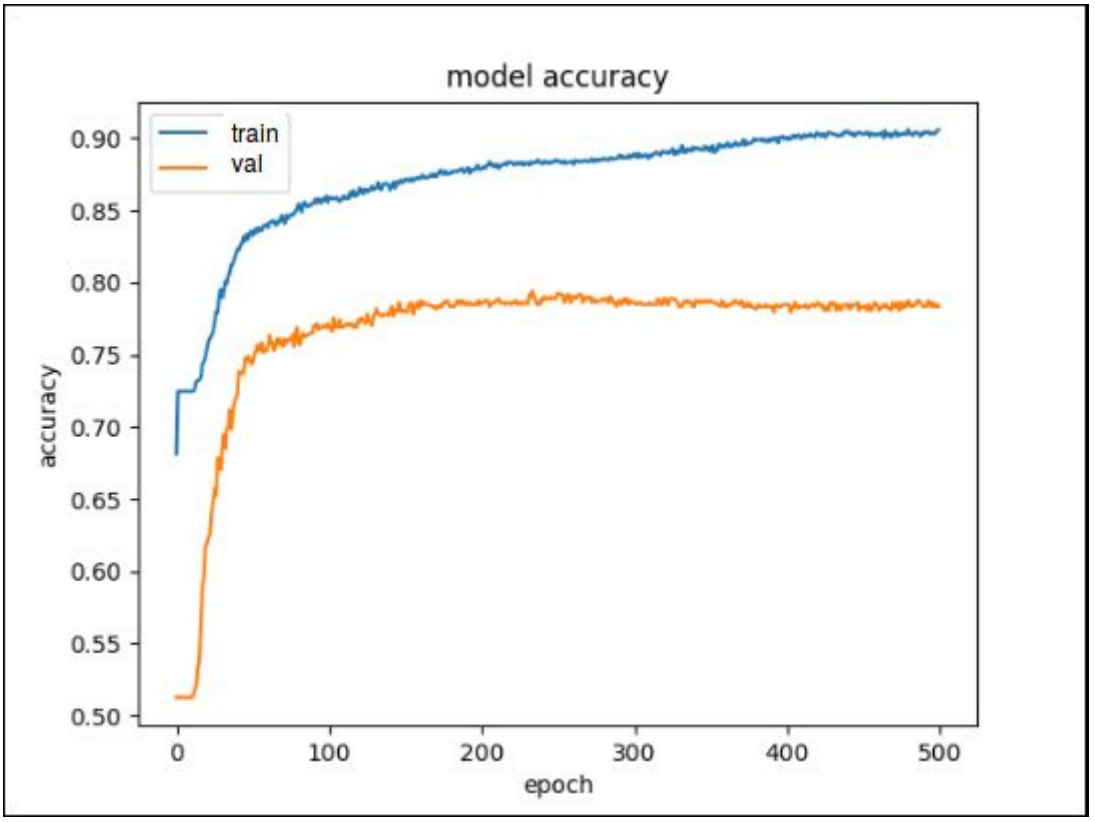


Figure 3.1. An attention model on top of a concatenation of word and aspect embeddings (Asp-Att model).



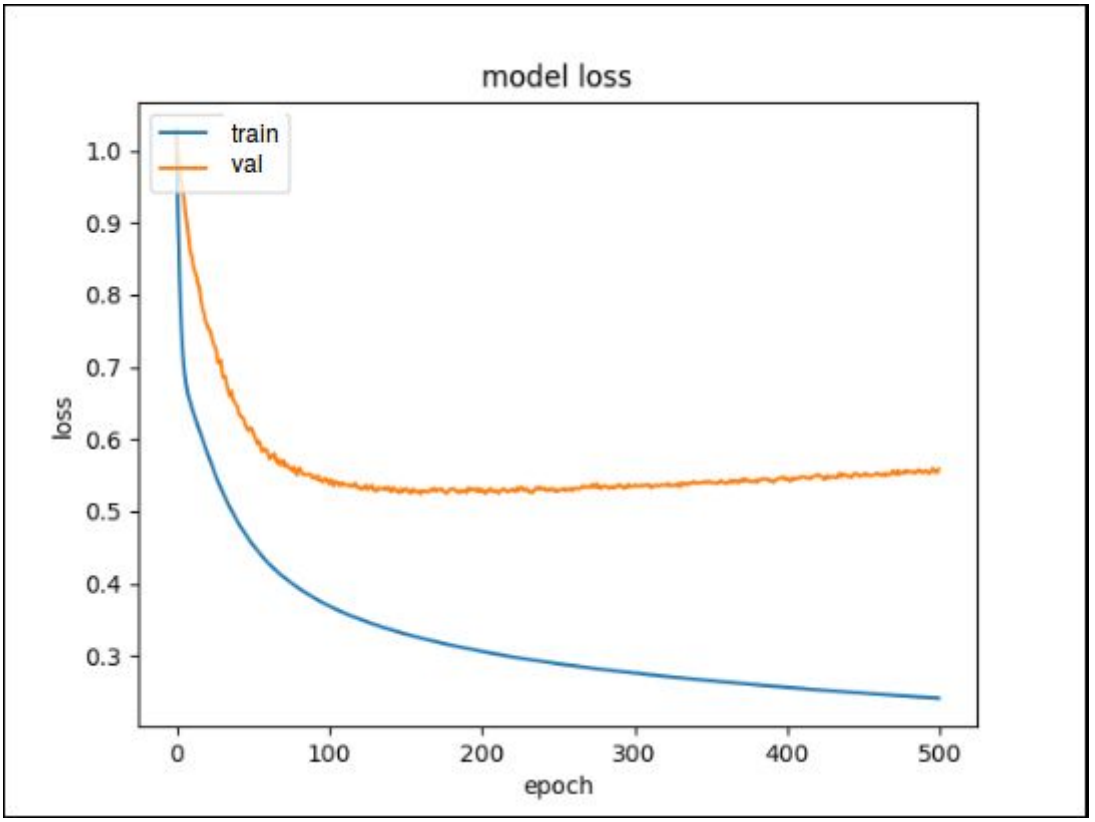


Figure 3.2. Evaluation of the Asp-Att model.

Improvement 1 :

In order to incorporate the information of preceding words, when examining each new word, a LSTM cell was used, and we added the attention mechanism on top of the hidden states. We call this model Asp-Att-LSTM. We can see a sudden increase in the loss after the 10th epoch probably due to overfitting (Figure 3.3). Moreover, both in the loss and in the accuracy diagram we can see a lot of big spikes on the validation data. This indicates that our model relies heavily on some specific instances of the training data and fails to generalize well.

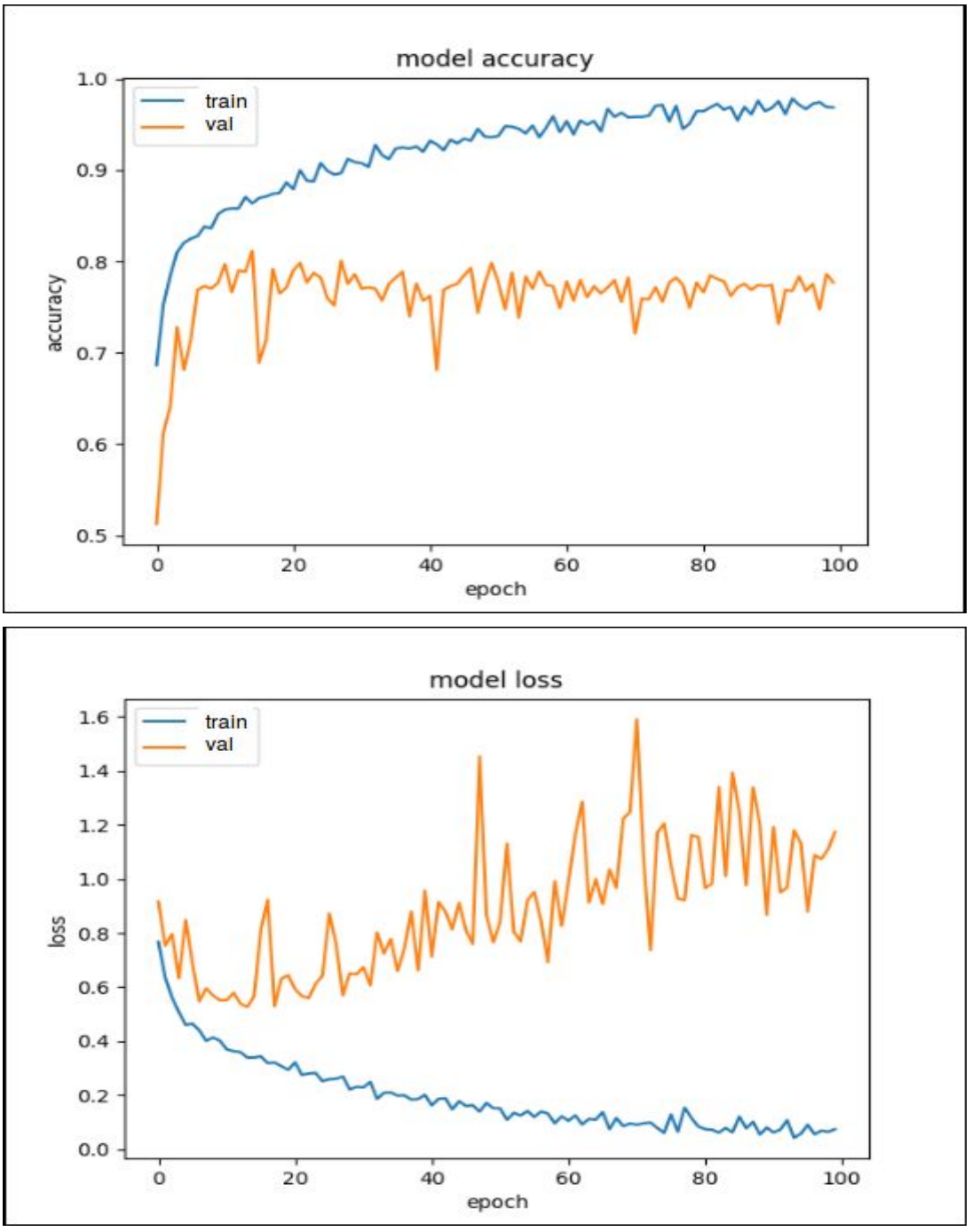


Figure 3.3. Evaluation of the Asp-Att-LSTM model.

Improvement 2:
In order to tackle the problem of overfitting we added dropout layers after the embeddings and before the final output. In both cases we kept the dropout mask fixed, meaning that in each sentence the dimensions that are dropped in the word embeddings are the same, for all the words of the sentence. We call this model Dr-Asp-Att-LSTM. As we can see at Figure 3.4 the spikes in both the loss and the accuracy are reduced significantly and both curves of the validation data seem to follow the training curves more smoothly. We can see that the optimum epoch to stop training would be the 30th, and we would achieve a loss of 0.52 and an accuracy score of 79%.

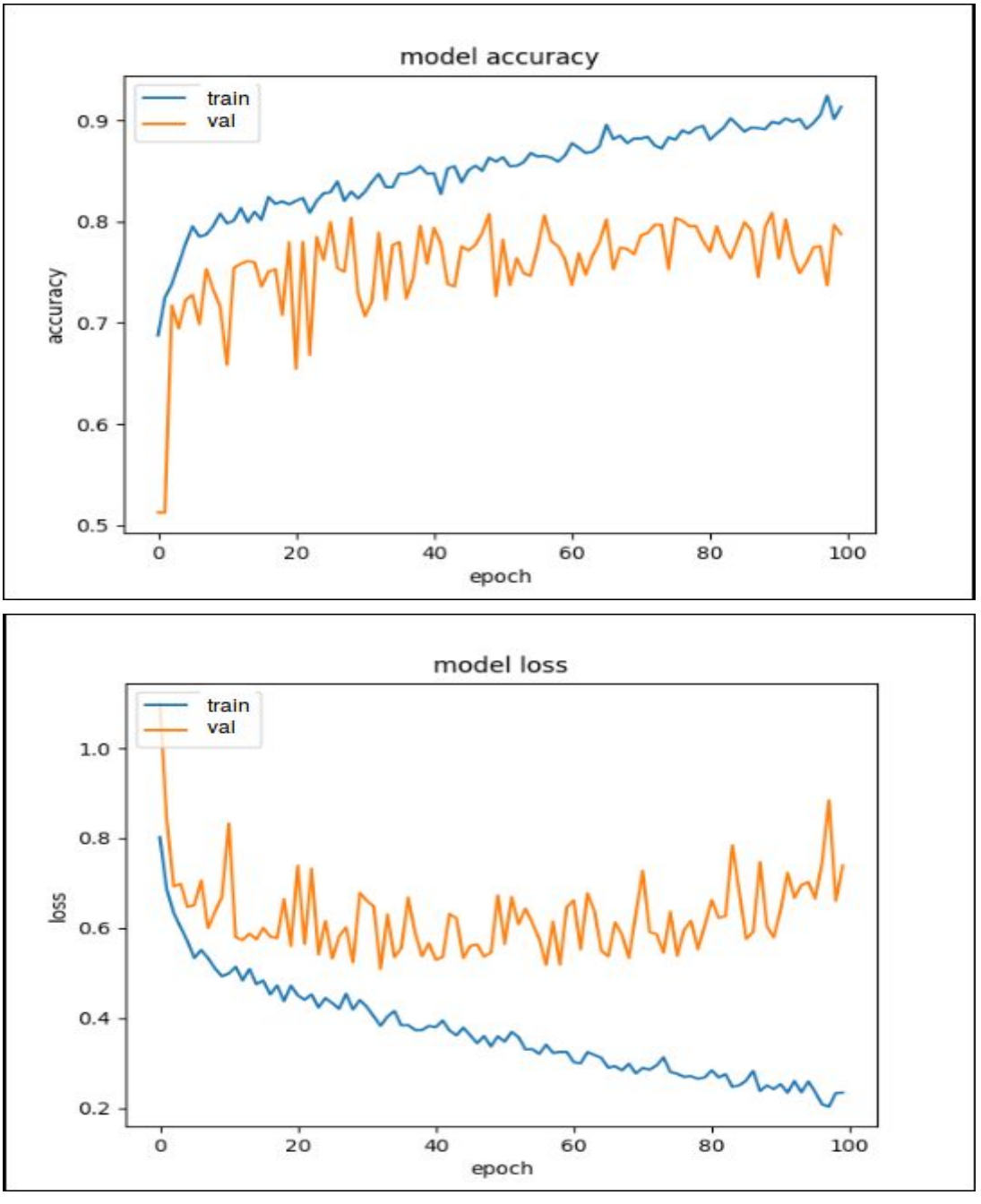


Figure 3.4. Evaluation of the Dr-Asp-Att-LSTM.

Improvement 3 :

Another addition to the model that was proposed by Wang et al. (2016) is to also concatenate the aspect embedding with the hidden states of the LSTM. This will make the information of the aspect better propagate through the system and directly interact with the attention mechanism (Figure 3.5)

We call this model Dr-HidAsp-Att-LSTM. As we can see at Figure 3.6, at the optimum epoch (40th) the loss is at 0.5 and the accuracy score is 80%. Although we see only a small improvement in loss and in accuracy we consider this change beneficial, because it also makes the system more stable.

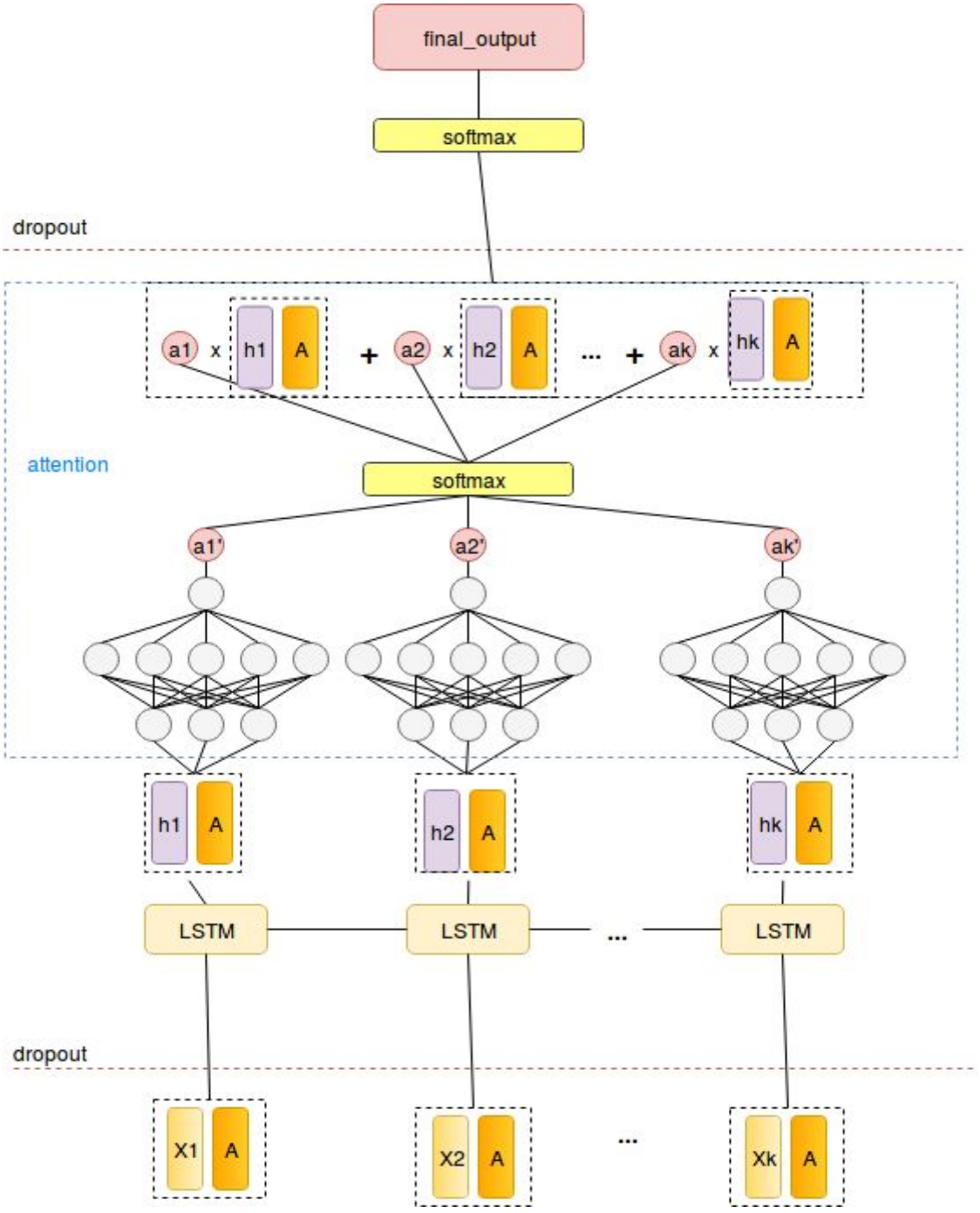


Figure 3.5.The Dr-HidAsp-Att-LSTM model.

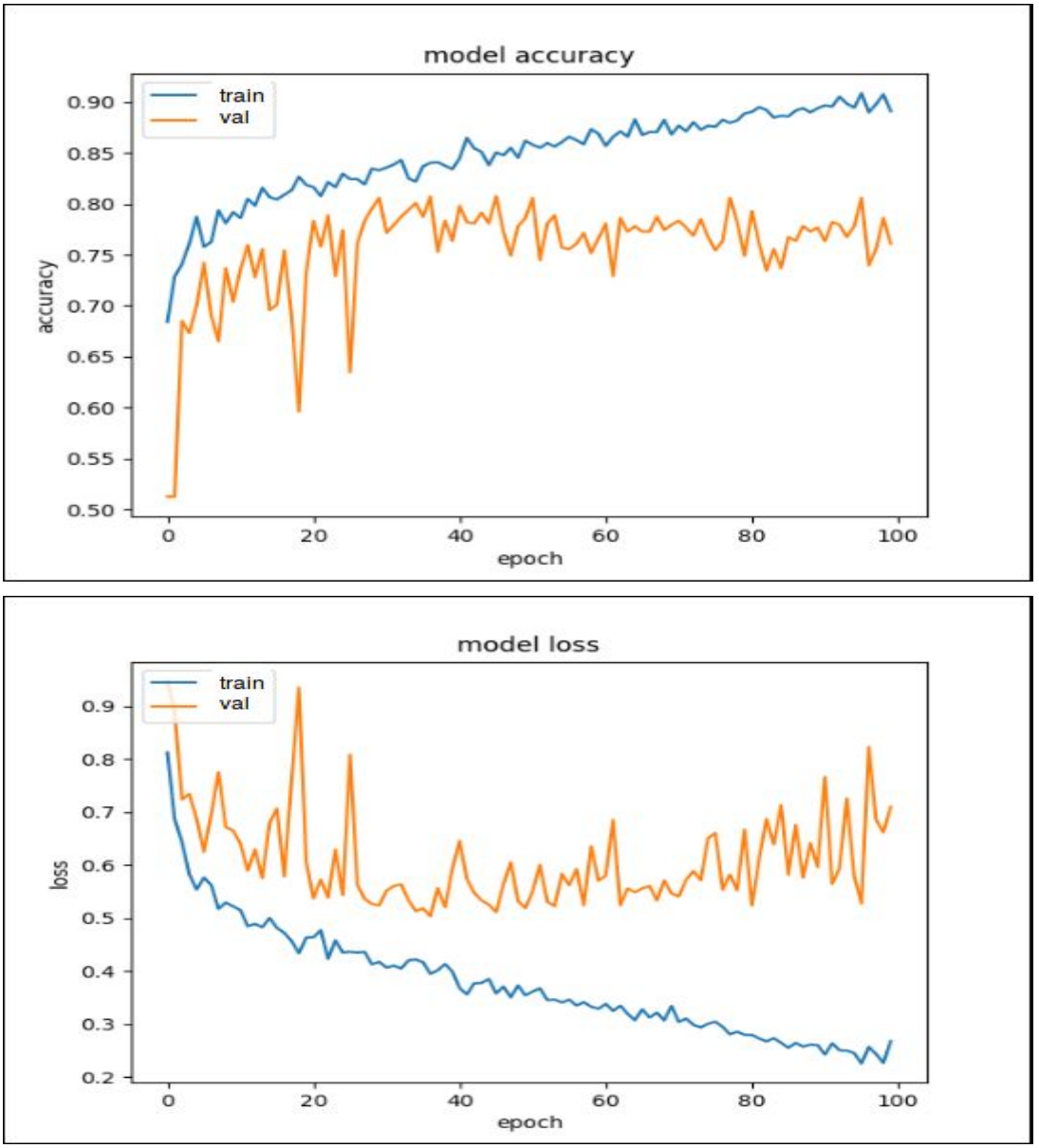


Figure 3.6. Evaluation of the Dr-HidAsp-Att-LSTM model.

Improvement 4 :

In a similar manner as in ACD, we add projection layers after the embeddings and before the final output. This will make us learn only the combination of dimensions that are deemed helpful for our decision. Given sentence with k word vectors (x_t) concatenated with a single aspect vector (A) $S = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_k\}$, where $\bar{x}_t = (x_t; A)$, $\bar{x}_t \in R^{d+m}$, $x_t \in R^d$, $A \in R^m$ we can define a projection layer that gives the following output for each $\bar{x}_t$.

$J = tanh\,(W_j \bar{x}_t + b_j)$, where $W_j \in R^{s\,x\,(d+m)}$, $b_j \in R^s$ and $s \ll (d+m)$

We use this layer in order to project each word vector from $(d+m)$ to s dimensions (s is much smaller than $(d+m)$) and at the end we concatenate all the projections of the word vectors into a

single flattened vector of size $(s \cdot k)$. We have to keep in mind that the same projection matrix is used for all the words of the sentence. In a similar manner works the other projection layer with the difference that it uses the activation function ReLU.We call this model Pr-Dr-HidAsp-Att-LSTM. As we can see at Figure 3.7 at the optimum epoch (40th) the loss is at 0.49 and the accuracy score is 80%. Moreover, we have almost eliminated the spikes in the accuracy diagram.

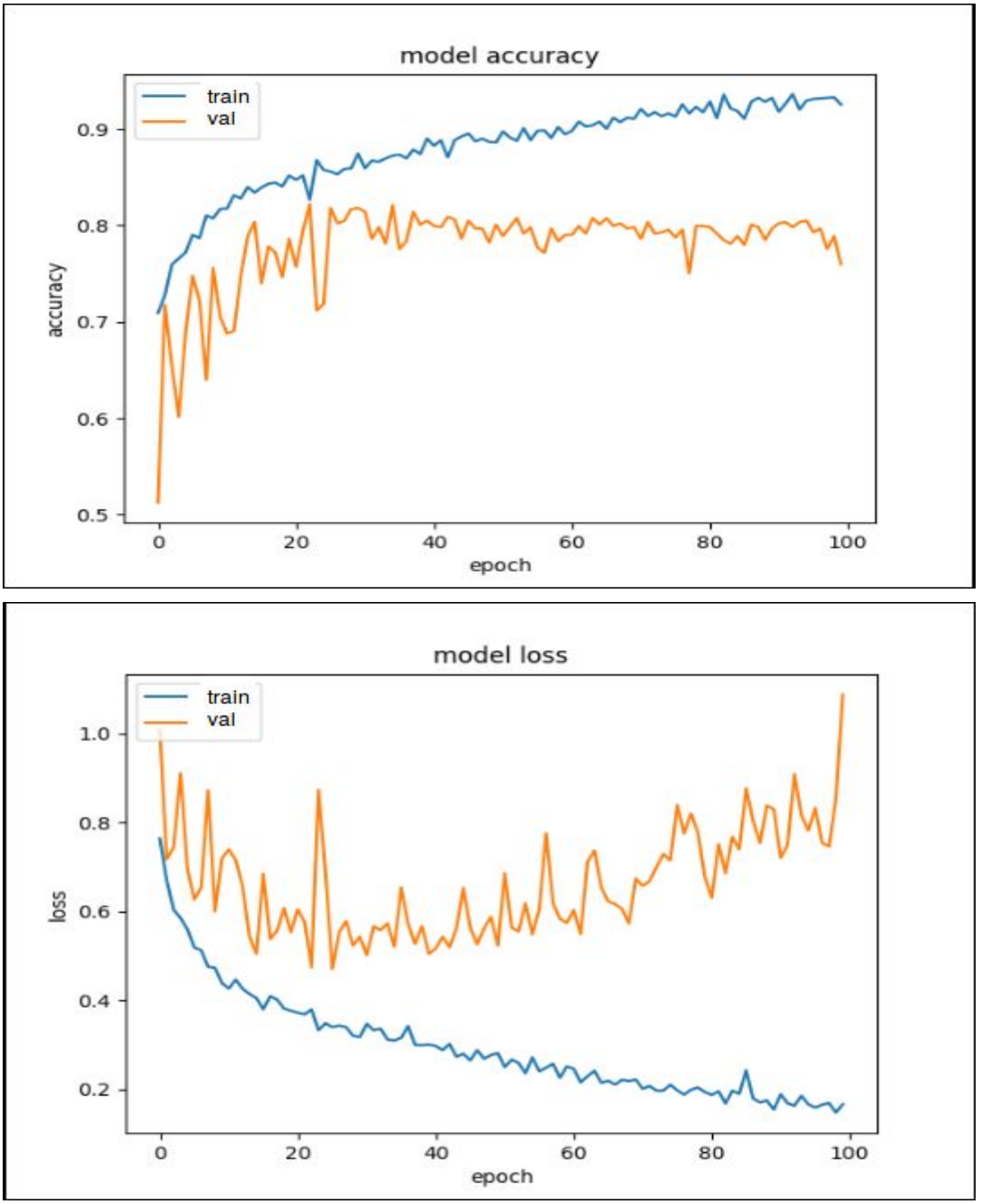


Figure 3.7. Evaluation of the Pr-Dr-HidAsp-Att-LSTM model.

Final Model :

As a final step we made the LSTM bidirectional, and we did hyperparameter tuning by randomly sampling different sets of hyperparameter and evaluating the performance on the validation data (Figure 3.8). We call this model Pr-Dr-HidAsp-Att-BiLSTM. The results Figure 3.9 indicate that

improvement is slight. At the optimum epoch (30th) the loss seems to be 0.49 and the accuracy score at 81%. The optimum hyper-parameters of this model are available at Appendix-Table B.

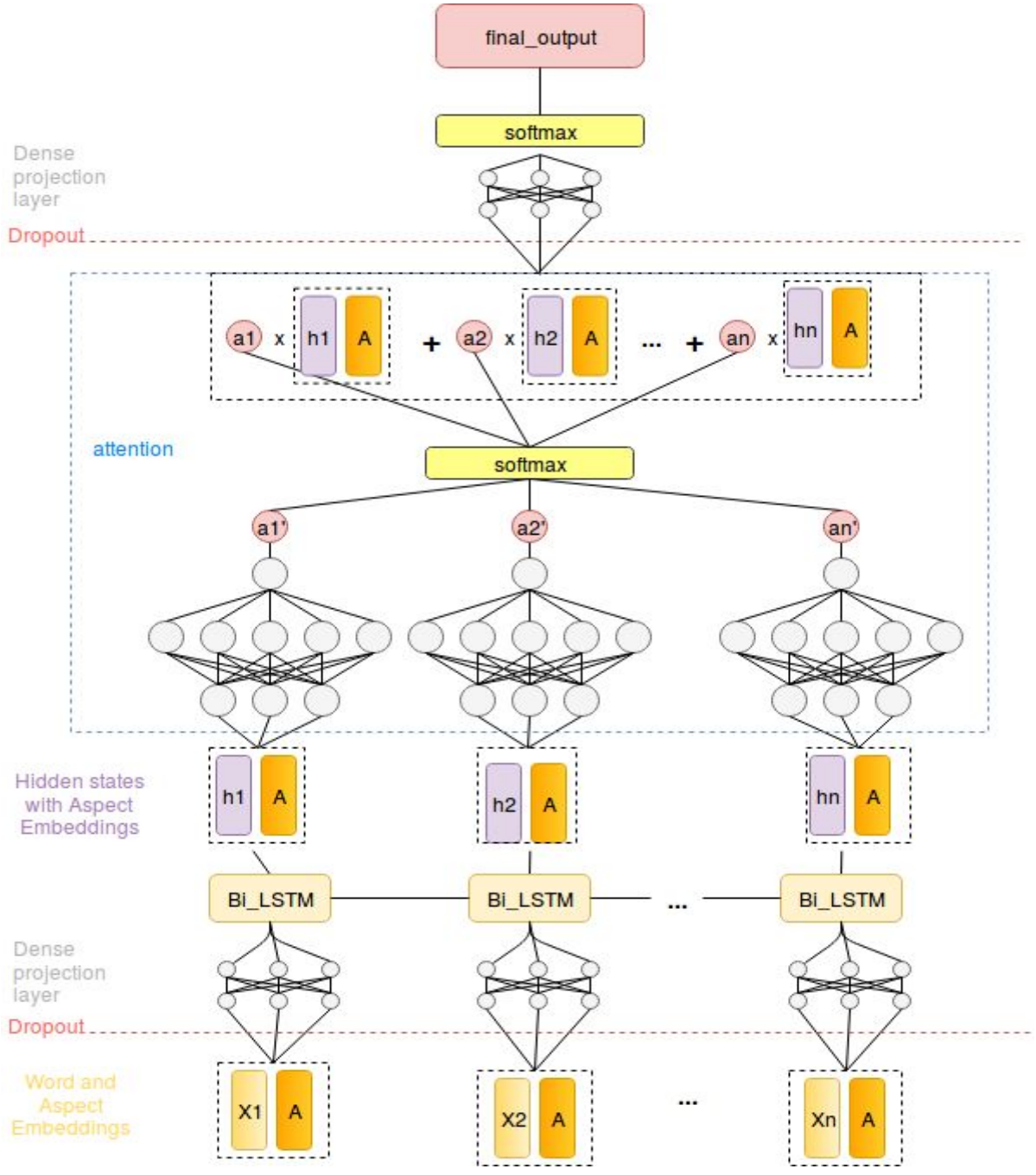


Figure 3.8. Final model (Pr-Dr-HidAsp-Att-BiLSTM).

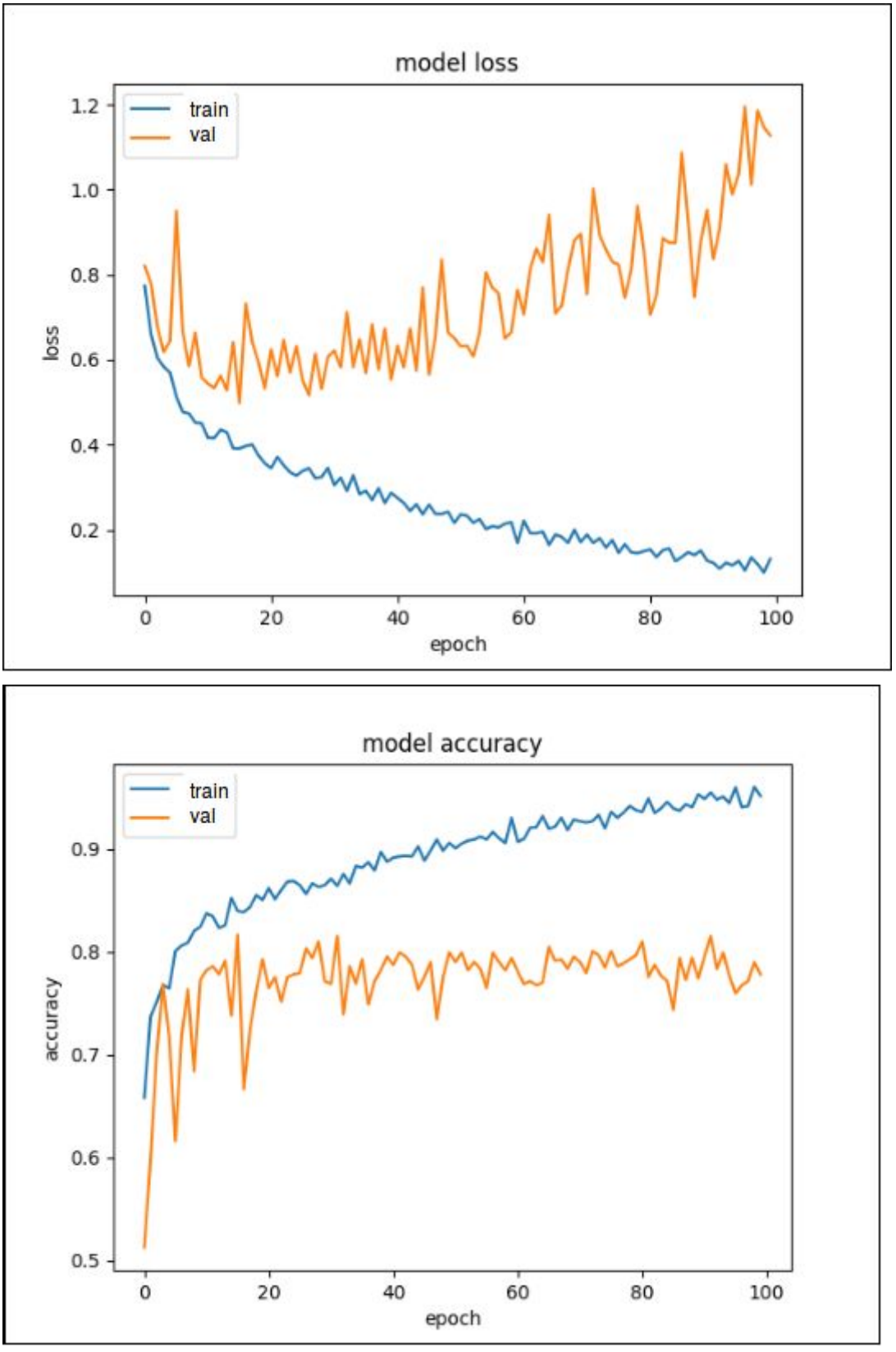


Figure 3.9. Evaluation of the Pr-Dr-HidAsp-Att-BiLSTM.

3.4.2. Data Augmentation using Neural Machine Translation

In order to create more instances similarly with the previous slot, each sentence was translated from English to German and back to English. In this manner we create a new sentence that holds the same meaning as the original one. In the previous slot we experimented with various settings of the neural machine translation (NMT) and came to the conclusion that the best performance is achieved by simply doubling the instances, by using only one middle language (German). This approach will be followed here as well. The new instances created via NMT are shown in the table below.

	Positive	**Negative**	**Neutral**	**Total**
Train (Original)	**1657**	**749**	**101**	**2507**
Train (using NMT)	**3314**	**1498**	**202**	**5014**

Table 6 : Data after using Neural Machine Translation in SP.

In order to evaluate our model we initially split the data (70% train and 30% validation) and then we augment only the training and not the validation data. This is because we wanted to have a validation set close in nature to the test dataset. By taking a look at the results of Figure 3.10, we can see that at the optimum epoch (30th), the validation loss is 0.4 and the accuracy score increases to 85%. The validation loss falls below the training loss and the validation accuracy is larger than the training accuracy. This is an expected result since the training dataset is much more noisy (due to augmentation) and is harder to make predictions on it.

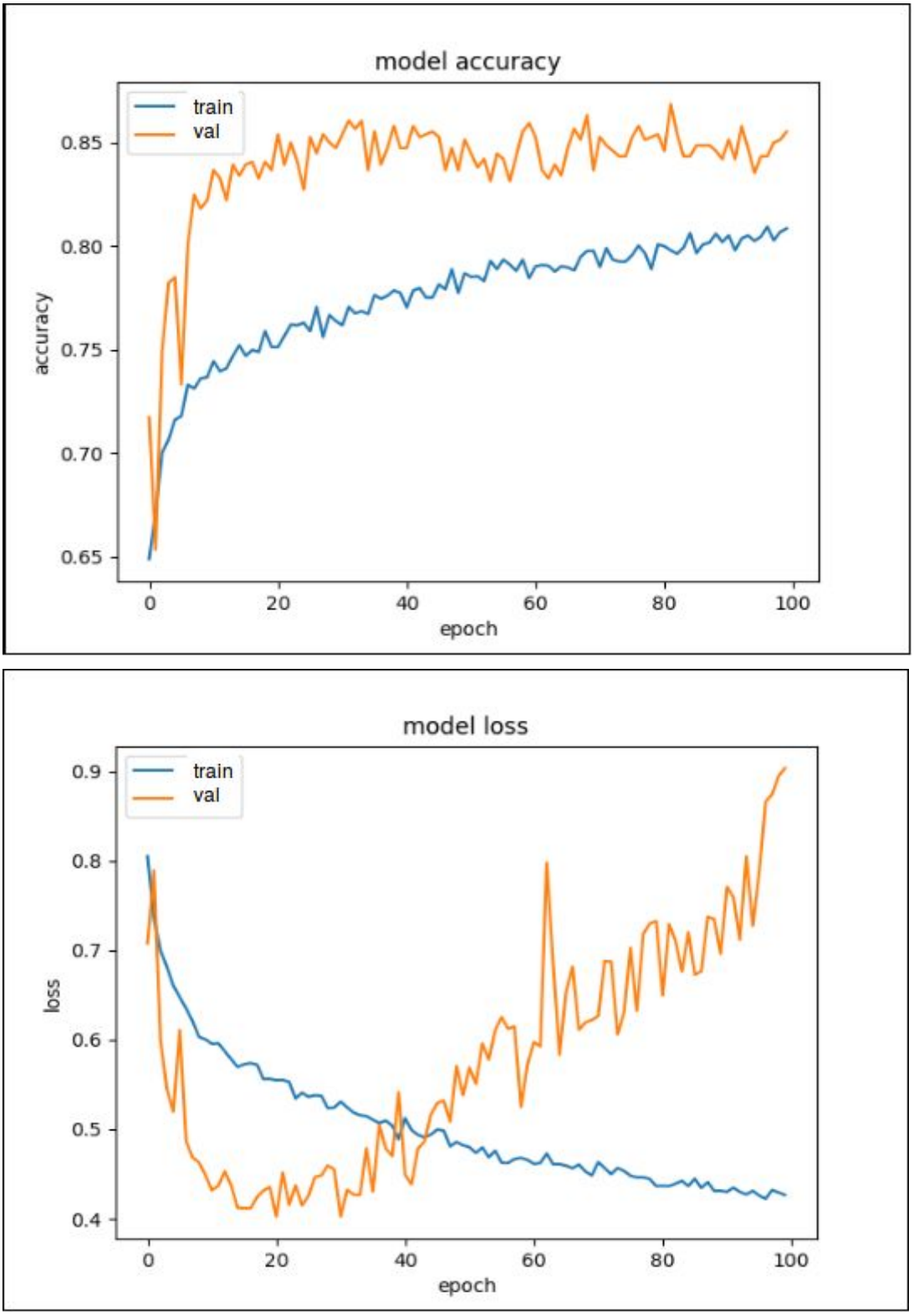


Figure 3.10. Evaluation of the Pr-Dr-HidAsp-Att-BiLSTM after NMT (Neural Machine Translation).

3.4.3. Final Results and Comparisons

After having selected and fine-tuned our model and implemented our augmentation technique, the last step was to evaluate our model. We evaluate against the baseline that was proposed by the competition (SVM) and the system that was implemented by the winning team of slot 3 at SemEval 2016 (Feedbacked Ensemble Modeling System). The results are shown in the table below. The Pr-Dr-HidAsp-Att-BiLSTM model achieves an accuracy score of 86.30%. Although we have not managed to outperform the best result of the competition, the current standing puts us in the best 3 results of the competition.

System	**Accuracy score**
Sem.2016 Slot 3 - Winner	**88.12**
SVM (Baseline model)	76.48
Bi-LSTM with aspect embeddings	86.30

Table 7 : Final Evaluation for the suggested model for SP.

4. Conclusions and Future Work

4.1. Conclusions

The systems that are proposed in this thesis not only achieve excellent results on ABSA data from the SemEval competition, but also present new approaches that have not been tested by other participants. To the best of our knowledge we have the best score in Entity Attribute Detection and one of the best 3 scores in Polarity Detection. We have proven that neural networks can achieve good results even in smaller datasets with the appropriate techniques. All the methods and experimentation of this thesis are available[3] for anyone who wants to experiment further with our systems. We do feel there is a good margin for further improvement. Moreover, the methods that were used for data augmentation using Neural Machine Translation are put together in a standalone library for anyone who wants to use them in NLP tasks[4].

During the implementation of the methods of this thesis more than a few challenges were faced. SemEval is a well-known competition with a good reputation that attracts a lot of people from the academic community. It is hard to suggest new approaches that have not already been tested, let alone achieve good results with them. Another big issue was the size of the training dataset. Only a few submissions in the competition used neural networks and the ones who did, have not had much success due to the small size of the dataset. This is why the augmentation techniques and the methods that helped the models generalize better were essential for both systems that were suggested.

4.2. Future Work

The following is a list with possible ideas in order to improve the current systems or even implement other systems. Some ideas that could be applied for both ACD and SP are:

a. Transfer learning: Use the weights from a similar system trained over a larger dataset (target-based sentiment analysis on Yelp dataset[5]) to initialize the weights of our systems up to the Bi-LSTM. This way we may avoid overfitting, and we will be able to transfer the knowledge from one task to a similar one.
b. Use FastText[6] in order to get better word embeddings. The advantage of this method is that we will get a good representation even for unknown words, by using a character level model.
c. Another way to get a better representation of unknown words is to categorize several words - symbols in one category and learn one embedding for all of them (e.g. we could replace all dates with a symbol and learn one embedding for all of them).

For the PR-G-Att-BiLSTM model suggested for ACD we could implement the following ideas:

[3] https://github.com/admdemiraj/Aspect-Based-Sentiment-Analysis
[4] https://github.com/admdemiraj/Text_Augmentation_via_Translation
[5] https://www.yelp.com/dataset/challenge
[6] https://fasttext.cc/

a. Instead of using one attention layer in order to make a decision for all categories, use one attention layer for each possible category. This way the attention layer will focus only on the words that are important for each category.
b. Instead of using a threshold of 0.5 in order to determine whether an aspect was present in the sentence or not, learn a different threshold for each category. This might be helpful because not the same level of certainty is necessary in order to determine if an aspect is present.

For the Pr-Dr-HidAsp-Att-BiLSTM model suggested for SP we could implement the following ideas:

a. In this slot we know the positions of the words that helped us decide which aspects are present. We could encode in the aspect embedding a distance measure from these words. The intuition here is that the sentiment word that characterizes an aspect is most likely located close to the aspect word. The main idea is to give the sentiment words that are close to the specific aspect words a higher score. Another extension of this idea is to give zero attention weight on the aspect word in order to let our model better focus on the sentiment word.
b. Finally, similarly with the suggestion on ACD we could create a different attention mechanism for each category (*positive,negative, neutral*).

References

[1] C. Brun, D. Popa, and C. Roux, "XRCE: Hybrid Classification for Aspect Based Sentiment Analysis". In Proceedings of the *8th International Workshop on Semantic Evaluation*, Dublin, Ireland, pp. 838–842, 2014.

[2] M. Chernyshevich, "IHS R & D Belarus : "Cross-domain Extraction of Product Features using Conditional Random Fields". *In Proceedings of the 8th International Workshop on Semantic Evaluation*, Dublin, Ireland, pp. 309–313, 2014.

[3] W. Jin and H. H. Ho, "A novel lexicalized HMM-based learning framework for web opinion mining". In Proceedings of the *26th Annual International Conference on Machine Learning*, Montreal, Canada, pp. 1–8, 2009.

[4] F. Li et al., "Structure-aware Review Mining and Summarization". In Proceedings of the *23rd International Conference on Computational Linguistics*, Beijing, China, pp. 653–661, 2010.

[5] Z. Toh and J. Su, "NLANGP: Supervised Machine Learning System for Aspect Category Classification and Opinion Target Extraction". In Proceedings of the *9th International Workshop on Semantic Evaluation*, Denver, Colorado, pp. 496–501, 2015.

[6] Z. Toh and J. Su, "NLANGP at SemEval-2016 Task 5: Improving Aspect Based Sentiment Analysis using Neural Network Features". In Proceedings of the *10th International Workshop on Semantic Evaluation*, San Diego, California, pp. 282–288, 2016.

[7] Z. Toh and W. Wang, "DLIREC: Aspect Term Extraction and Term Polarity Classification System". In Proceedings of the *8th International Workshop on Semantic Evaluation*, SemEval 2014, Dublin, Ireland, pp. 235–240, 2014.

[8] W. Wang, S. J. Pan, D. Dahlmeier, and X. Xiao, "Recursive Neural Conditional Random Fields for Aspect-based Sentiment Analysis". In Proceedings of the *2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, pp. 616–626, 2016.

[9] C. Brun, J. Perez, C. Roux, and C. Europe, "XRCE at SemEval-2016 Task 5: Feedbacked Ensemble Modelling on Syntactico-Semantic Knowledge for Aspect Based Sentiment Analysis". *In Proceedings of the 10th International Workshop on Semantic Evaluation*, San Diego, California pp. 277–281, 2016.

[10] A. García-Pablos, M. Cuadros, S. Gaines, and G. Rigau, "Unsupervised acquisition of domain aspect terms for aspect based opinion mining". In the *Natural Language*, vol. 53, pp. 121–128, 2014.

[11] A. Giannakopoulos, C. Musat, A. Hossmann, and M. Baeriswyl, "Unsupervised Aspect Term Extraction with B-LSTM & CRF using Automatically Labelled Datasets". In Proceedings of the *8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, Copenhagen, Denmark, pp. 180–188, 2017.

[12] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier, "An Unsupervised Neural Attention Model for Aspect Extraction". In Proceedings of the *55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada, pp. 388–397, 2017.

[13] Q. Liu, Z. Gao, B. Liu, and Y. Zhang, "Automated rule selection for aspect extraction in opinion mining". In Proceedings of the *24th International Joint Conference on Artificial Intelligence*, Buenos Aires, Argentina, vol. 2015, pp. 1291–1297, 2015.

[14] M. Zhang, Y. Zhang, and D. Vo, "Gated Neural Networks for Targeted Sentiment Analysis". In Proceedings of the *13th AAAI Conference on Artificial Intelligence*, Phoenix, Arizona, pp. 3087–3093, 2016.

[15] Y. Wang, M. Huang, L. Zhao, and X. Zhu, "Attention-based LSTM for Aspect-level Sentiment Classification". In Proceedings of the *2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, pp. 606–615, 2016.

[16] D. T. Vo and Y. Zhang, "Target-dependent twitter sentiment classification with rich automatic features". In Proceedings of the *Twenty-Fourth International Joint Conference on Artificial Intelligence*, Buenos Aires, Argentina, pp. 1347–1353, 2015.

[17] D. Tang, B. Qin, and T. Liu, "Aspect Level Sentiment Classification with Deep Memory Network". In Proceedings of the *2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, pp. 214–224, 2016.

[18] D. Tang, B. Qin, X. Feng, and T. Liu, "Effective LSTMs for Target-Dependent Sentiment Classification". In Proceedings of *the 26th International Conference on Computational Linguistics*, Osaka, Japan, pp. 3298–3307, 2015.

[19] L. Dong, F. Wei, C. Tan, D. Tang, M. Zhou, and K. Xu, "Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification". In Proceedings of the *52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, Maryland, pp. 49–54, 2014.

[20] P. Chen, Z. Sun, L. Bing, and W. Yang, "Recurrent Attention Network on Memory for Aspect Sentiment Analysis". In Proceedings of the *2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, pp. 452–461, 2017.

[21] S. Ruder, P. Ghaffari, and J. G. Breslin, "A Hierarchical Model of Reviews for Aspect-based Sentiment Analysis". In Proceedings of the *2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, pp. 999–1005, 2016.

[22] J. Liu and Y. Zhang, "Attention Modeling for Targeted Sentiment". In Proceedings of the *15th Conference of the European Chapter of the Association for Computational Linguistics*, Valencia, Spain, pp. 572–577, 2017.

[23] D. Ma, S. Li, X. Zhang, and H. Wang, "Interactive attention networks for aspect-level sentiment classification". In Proceedings of the *2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, pp. 4068–4074, 2017.

[24] C. Brun, J. Perez, C. Roux, "XRCE at SemEval-2016 Task 5: Feedbacked Ensemble Modelling on Syntactico-Semantic Knowledge for Aspect Based Sentiment Analysis". In Proceedings of the *10th International Workshop on Semantic Evaluation*, San Diego, California, pp. 277–281, 2016.

[25] T. Devries and G. W. Taylor, "Dataset augmentation in feature space". In the Proceedings of the *5th International Conference on Learning Representations*, Toulon, France, pp. 1–12, 2017.

[26] N. Chawla and K. Bowyer, "SMOTE: Synthetic Minority Over-sampling Technique". In the *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[27] D. F. Specht, "Probabilistic neural networks". In *Neural Networks*, vol. 3, no. 1, pp. 109–118, 1990.

[28] H. Zou and T. Hastie, "Regularization and variable selection via the elastic-net". *In Journal of the Royal Statistical Society*, vol. 67, no. 2, pp. 301–320, 2005.

[29] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory". *In Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[30] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. AL-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq, V. Hoste, M. Apidianaki, X. Tannier, N. Loukachevitch, Evgeniy Kotelnikov, Núria Bel, S. M. Jiménez-Zafra and G. Eryiğit, "SemEval-2016 Task 5: Aspect Based Sentiment Analysis". In Proceedings of the *10th International Workshop on Semantic Evaluation*, San Diego, California, pp. 342–349, 2016.

[31] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar and I. Androutsopoulos, "SemEval-2015 Task 12: Aspect Based Sentiment Analysis". In Proceedings of the *9th International Workshop on Semantic Evaluation*, Denver, Colorado, pp. 486–495, 2015.

[32] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos and S. Manandhar, "SemEval-2014 Task 4: Aspect Based Sentiment Analysis". In Proceedings of the *8th International Workshop on Semantic Evaluation*, Dublin, Ireland, pp. 27–35, 2014.

[33] V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[34] J. Lafferty, A. McCallum and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data". *I*n the Proceedings of the *18th International Conference on Machine Learning,* San Francisco, California, pp. 282–289, 2001.

[35] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space". In the Proceedings of the *International Conference on Learning Representations,* Scottsdale, Arizona, pp. 1–12, 2013.

[36] L. J. P. Van Der Maaten and G. E. Hinton, "Visualizing high-dimensional data using t-sne". In *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[37] J. Pavlopoulos, P. Malakasiotis, and I. Androutsopoulos, "Deeper Attention to Abusive User Content Moderation". In Proceedings of the *2017 Conference on Empirical Methods in Natural Language*, Copenhagen, Denmark, pp. 1136–1146, 2017.

[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In Journal of *Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

Appendix

Word embeddings	300 dimensions pretrained
Gaussian noise	0,2
Dropout layers	0,3 / 0,2
Dense layers	1024 activation Tanh / 250 activation ReLU
Bi-LSTM	256
Activation function for final decision	Sigmoid (one sigmoid for each category)
Optimizer	RMSprop
Loss	Binary Cross Entropy (for each category)

Table A: optimum hyper-parameters for the PR-G-Att-BiLSTM model (Slot 1).

Word embeddings	300 dimensions pretrained
Aspect embeddings	50 dimensions
Dropout layers	0,2 / 0,2
Dense layers	1024 activation Tanh / 256 activation ReLU
Bi-LSTM	256
Activation function for final decision	Softmax
Optimizer	RMSprop
Loss	Categorical Cross Entropy

Table B: optimum hyper-parameters for the Pr-Dr-HidAsp-Att-BiLSTM model (Slot 3).