Applications of Pre-trained Language Models in Sentiment and Authorship Tasks

by
Yifan Zhang

A dissertation submitted to the Department of Computer Science,
College of Natural Sciences and Mathematics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Computer Science

Chair of Committee: Arjun Mukherjee

Committee Member: Omprakash Gnawali

Committee Member: Panruo Wu

Committee Member: Xuqing Wu

University of Houston
May 2021

ABSTRACT

In recent years, transfer learning in the form of pre-trained neural language models (LMs) has significantly transformed the research and application of natural language processing (NLP). Techniques and models, such as ELMo, ULMFit, Transformer, and BERT, have claimed state-of-the-art results on a wide range of NLP tasks. This dissertation explores utilizing pre-trained LMs to address two major areas in NLP: Authorship Verification and Sentiment Analysis. Our design focus for these models is to achieve not only state-of-the-art performance in the respective tasks, but also high flexibility and high interpretability.

For the Authorship Verification problem, we propose an unsupervised solution that utilizes pre-trained deep language models to compute a new metric called *DV-Distance*. The proposed metric is a measure of the difference between the two authors compared against pre-trained LMs. Our design addresses the problem of non-comparability in authorship verification, frequently encountered in small or cross-domain corpora. To the best of our knowledge, this work is the first one to introduce a method designed with non-comparability in mind from the ground up, rather than indirectly. It is also one of the first to use deep language models in this setting. The approach is intuitive, and it is easy to understand and interpret through visualization. Experiments on four datasets show our methods matching or surpassing current state-of-the-art and strong baselines in most tasks.

For sentiment analysis, we propose two iterations of a framework called the sentiment-aspect Attribution Module (SAAM) . SAAM works on top of traditional neural networks and is designed to address the problem of multi-aspect sentiment classification and sentiment regression. The framework works by exploiting the correlations between sentence-level embedding features and variations of document-level aspect rating scores.

We first propose several variations of SAAM and demonstrate their effectiveness on top of CNN and RNN based models. Experiments on a hotel review dataset and a beer review dataset have shown that SAAM can improve the sentiment analysis performance over corresponding base models. Moreover, because of how our framework intuitively combines sentence-level scores into document-level scores, it can provide a deeper insight into data (e.g., semi-supervised sentence aspect labeling). Hence, we also provide a detailed analysis that shows the potential of our models

for other applications, such as sentiment snippet extraction.

Lastly, this dissertation also presents SAAM v2. SAAM v2 dramatic improvement over the original version, by addressing three of its significant shortcomings. We demonstrate SAAM v2's capabilities by combining it with pre-trained language model architectures AWD-LSTM and RoBERTa. The evaluation of SAAM v2 on the hotel and beer review datasets confirms that the module can provide better expressiveness and overall performance. Furthermore, the model can estimate sentence-level aspects at a much higher accuracy. We end our model analysis by showcasing some of the fine-grained latent information discovered by SAAM v2.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

In recent years, transfer learning in the form of pre-trained neural language models (LMs) has significantly transformed the research and application of natural language processing (NLP). Techniques and models, such as ELMo, ULMFit, Transformer, and BERT have claimed state-of-the-art results on a wide range of NLP tasks. These tasks include, but are not limited to, document classification, such as sentiment analysis (IMDb dataset [44], Yelp review dataset compiled by [78], word level sequence labeling, structured prediction such as parsing, text generation such as summarization and question answering. Indeed, it may not be an exaggeration to say that the whole NLP research field has experienced a revolution since the first successful application of pre-trained LMs was proposed in 2018.

With the fast-paced advancement of language modeling techniques, modern LMs have become incredibly complex and large. For example, the full-sized variant of the BERT model consists of 24 layers and 345 million parameters. A more extreme example would be Turning-NLG [49], a model with 78 transformer layers and 17 billion parameters, which was also shown to be exceptionally strong at language modeling and other NLP tasks.

However, despite the massive amount of interest and effort afforded to the LMs, little research has been conducted to better utilize token-level LM outputs to arrive at the ultimate document-level predictions. Less formally, we refer to methods or models that take in and utilize token-level vectors from LMs to make document-level predictions as token-to-doc connections. Token-to-doc connections are prevalent and exist in all forms of document classification models, including sentiment classification, question classification, and topic classification. Despite their importance, as far as we know, almost all existing document classification models have a token-to-doc connection based on some combinations of max-pooling, average-pooling, and a few fully connected layers.

To advance the study of this under-explored field, in this dissertation, we will propose four novel methods and models that play the role of token-to-doc connections. The application will

revolve around two vital and challenging NLP tasks: Authorship Verification (AV) and Multi-aspect Sentiment Analysis (MASA). It is worth noting that the methods and models we proposed are not limited to solving just these two types of problems. There remains much potential for our models to be applied to other tasks; and we will expand on this later in the corresponding sections.

1.1 Authorship Verification

Authorship Attribution (AA) [64] and Verification (AV) [43] are critical, challenging problems in this age of "fake news". The former attempts to identify *who* wrote a specific document; the latter concerns itself with the problem of finding out whether the same person authored several documents or not. Ultimately, the goal of AV is to determine whether the same author wrote any two documents of arbitrary authorship. These problems have attracted renewed attention, as we urgently need better tools to combat content farming, social bots, and other forms of communication -pollution.

An interesting aspect of authorship problems is that technology used elsewhere in NLP has not yet penetrated it. Up until the very recent PAN 2018 and PAN 2020 authorship events [30, 4], the most popular and effective approaches still largely rely on n-gram features and traditional machine learning classifiers, such as support vector machines (SVM) [11] and trees [15]. Elsewhere, these methods were recently overshadowed by deep neural networks. This phenomenon may be primarily attributed to the fact that authorship problems are often data constrained — as the amount of text from a particular author is often limited. From what we know, only a few deep learning models have been proposed and shown to be effective in authorship tasks [1, 20, 6], and these networks require a good amount of text to perform well. Likewise, transfer learning may not have been utilized to its full potential, as some of the recent work in deep language models shows it to be a silver bullet for tasks lacking training data [22].

In Chapter 2, we propose two deep neural language model based AV methods: *DV-Distance* and *DV-Projection*. Both methods are built upon the idea of estimating the magnitude and the direction of deviation of a document from the *normal writing style* (NWS), where the NWS is

modeled by state-of-the-art language models such as the AWD-LSTM and RoBERTa architecture introduced in [48, 40].

Among the two methods proposed, DV-Distance is a fully unsupervised method that directly reflects the magnitude and direction of the writing style deviation. Despite not requiring any ground truth information, DV-Distance can out-perform many previous state-of-art methods by a large margin. Based on the concept and framework of DV-Distance, we also proposed a supervised neural architecture, DV-Projection. DV-Projection works as a token-to-doc connection module by projecting deviation vectors (DVs) into a separate space and then subsequently combining them to form a document level prediction. We theorize that doing so allows the model to identify dimensions that are more appropriate for authorship tasks. Experiments show that this architecture's performance is significantly better compared to DV-Distance on academic and formal writing datasets.

In addition to the above two methods constructed based on the concept of DVs, we also propose a more conventional end-to-end Siamese neural network architecture named 2WD-UAV. 2WD-UAV is built on top of a multi-layer pre-trained LSTM language model and trained with various regularization techniques, such as adversarial noise injection. Experiment results prove the model to be very competitive in several author verification datasets. We will mainly use this model as a strong baseline to compare against.

Both DV-Distance and DV-Projection have intuitive and theoretically sound architecture and come with elegant interpretability. Moreover, both proposed methods contain previously unexplored token-to-doc connection techniques, which utilize outputs of the pre-trained LM to inform document-level predictions. We end this chapter with visualizations of DVs for document pairs from the same and different authors, partially verifying our initial hypothesis.

The work in Chapter 2 primarily relates to the following peer-reviewed articles:

- **Zhang, Y.**, Boumber, D., Hosseinia, M., Yang, F. and Mukherjee, A., 2021. Improving authorship verification using linguistic divergence. In *Workshop on Reducing Online Misinformation through Credible Information Retrieval (ROMCIR 2021), held as part of ECIR*

2021.

- Boumber, D., **Zhang, Y.**, Hosseinia, M., Mukherjee, A. and Vilalta, R., 2019. Robust authorship verification with transfer learning. In *Proceedings of the 2020 International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2020).*

The following publication is related, but will not be extensively discussed in Chapter 2:

- Boumber, D., **Zhang, Y.** and Mukherjee, A., 2018. Experiments with convolutional neural networks for multi-label authorship attribution. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).*

1.2 Multi-aspect Sentiment Analysis

Aspect level sentiment analysis is one of the main paradigms of sentiment analysis [54]. Extracting sentiments on each aspect of the entity is a critical tasks under aspect-level sentiment analysis. While there have been several works in aspect extraction [23, 58], opinion term extraction [12], aspect based summarization [23, 70, 81], joint aspect and sentiment models [26, 39, 53, 80], and polarity identification [63], the task of MASA has received limited attention.MASA refers to extracting each aspect's sentiment value (in polarity or rating score) at the document level. To our knowledge, the most closely related works on MASA are those in [41, 62, 70, 72]. However, these works either rely on aspect keyword supervision or only produce an aspect-based summary of an entity using latent topic attribution as opposed to the proposed fine-grained MASA at the sentence level. The individual differences between these works and ours are detailed in Section 3.1.

In Chapter 3 of this dissertation, we propose a novel neural network-based framework called the sentiment-aspect attribution module (SAAM) to solve the problem of document level MASA. The proposed SAAM module can be trained using a set of documents tagged with overall and aspect ratings. During inference, SAAM employs the *latent sentiment-aspect attribution* (LSAA) mechanism, where it assigns a latent aspect distribution to each sentence and estimates their sentiment scores. The estimated latent aspect distribution and sentiment scores for each sentence

• Definitely not a 5 star resort I'm dumbfounded that this hotel gets good reviews and is so highly rated. [1.23, Value] • It's decidedly a 3 star property, not 5 stars as indicated. [-0.04, Service] • The rooms are very dated and run down, old crappy beds and pillows, an old tv and overall poorly maintained. [-2.97, Room] • The whole property is pretty run down and old–looking. [-0.47, Location] • The food is subpar, not one meal I had would be called great. [-2.23, Service] • The service is uneven and the staff is poorly trained and uninformed. [-2.23, Service] • The beach is great, it's the only redeeming factor. [1.27, Location] • However the resort is a 1-hour taxi trip from the airport. [1.68, Location]

Overall:	★★☆☆☆	Value:	★☆☆☆☆
Room:	★☆☆☆☆	Location:	★★★★☆
Cleanliness:	★★☆☆☆	Service:	★★☆☆☆

Figure 1: A sample hotel review with user-submitted ratings shown beneath. Sentiment scores and aspects assigned to sentences by our model in brackets.

of a document are then pooled together to estimate the document-level review ratings. We proposed three variations of our SAAM framework (two classifications and one regression) to demonstrate the possibilities available.

Chapter 4 of this dissertation proposes a greatly improved version of SAAM named SAAM v2. The new model improves on the existing one by addressing three critical issues. Consequently, SAAM v2 demonstrates better performance at both document-level sentiment estimation and sentence level aspect estimation.

To our knowledge, the proposed models are the first neural network models capable of discovering both sentiment and aspect information at the sentence level with only document-level aspect-rating labels. Moreover, the frameworks we introduced in this work are not independent, specific neural network architectures. Instead, they are add-on "token-to-doc" components that can be added to other popular neural network architectures to support MASA and LSAA. As will be detailed in Chapter 3, we showcase the proposed SAAM framework by stacking three variations of SAAM on top of a CNN [31] and a GRU-based RNN to demonstrate the framework's ability to generalize, and we compare the performance between them. In Chapter 4, we showcase SAAM v2 by stacking it on top of multi-layer pre-trained LSTM and RoBERTa base models.

Experimental results on the TripAdvisor hotel review dataset and BeerAdvocate beer review dataset illustrate the effectiveness of the proposed approaches by showing performance improvement over corresponding base models, as well as other baselines on several metrics for classification and

• This beer is yellow, fizzy, and clearly meant for washing dirt out of your mouth after mowing the lawn. `[1.035, Appearance]` • I'm not even sure it's good for that. `[3.245, Taste]` • It's definitely yellow and fizzy, with no head to speak of, and zero lacing. `[-1.27, Appearance]` • It almost smells like a loaf of bread, and nearly tastes the same. `[4.255, Aroma]` • It's very earthy and grainy with nary a hop to be found. `[3.58, Taste]` • Man, I love me some Caldera, but I would rather drink a Bud Light than this on a hot summer day. `[1.845, Appearance]` • Sorry guys, but this beer gets an F. `[1.495, Taste]`

Overall: ★☆☆☆☆

Appearance:	★☆☆☆☆	Taste:	★☆☆☆☆
Palate:	★☆☆☆☆	Aroma:	★½☆☆☆

Figure 2: A sample beer review with user-submitted ratings shown beneath. Sentiment scores and aspects assigned to sentences by our model in brackets.

regression variations of the MASA task. Additionally, we evaluate our model's ability to attribute aspect labels to each sentence of a document by using manually labeled data and a heuristic keyword approach. We publish these processed datasets and sentence aspect labeling to better promote research in this novel task.

The work in Chapter 3 primarily relates to the following peer-reviewed article:

- **Zhang, Y.**, Yang, F., Hosseinia, M. and Mukherjee, A., 2020. Multi-aspect Sentiment Analysis with Latent Sentiment-Aspect Attribution. In *The 2020 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2020).*

The work in Chapter 4 primarily relates to the following article, which has been submitted to EMNLP 2021 and is currently pending review:

- **Zhang, Y.**, M., Mukherjee, 2019. Multi-aspect Sentiment Analysis with Improved Sentiment-Aspect Attribution Module. Submitted *The 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020).*

2 Authorship Verification

2.1 Problem Description

In the following sections, we use the symbol P to denote an AV problem. Each problem P consists of two elements: a set of *known documents* K, and *unknown documents*, U. Similarly, k and

u represent a single known and unknown document, respectively. The task is then to find a hypothesis, h, that takes in both components and correctly estimates the probability that the same author writes them. Important in many forensic, academic, and other scenarios, AV tasks remain very challenging due to several reasons. For one, in a cross-domain AV problem, the documents in K and u could be of entirely different genres and types. More specifically, K could contain several novels written by a known author, while u could be a Twitter post. Another example demonstrating why a cross-domain model may be necessary is the case of a death note [65], as it is implausible to obtain K containing death notes written by the suspect. Furthermore, solving an AV problem usually involves addressing one or more types of limited training data challenges: a limited amount of training problems P, out-of-set documents and authors appearing in test data, or a limited amount of content in the document sets $\{K, U\}$ of a particular problem P. Many methods use sophisticated forms of test-time processing, data augmentation, or ensembling to successfully minimize these challenges' impact and achieve state-of-the-art results [1, 5]. However, such solutions typically result in prohibitively slow performance, most require a considerable amount of tuning, and almost all of them, to the best of our knowledge, require labeled data. As a result, existing methods are not relevant in many real-world scenarios.

k: I suppose that was the reason. We were waiting for you without knowing it. Hallo!
u: He maketh me to lie down in green pastures; he leadeth me beside the still waters.

Figure 3: Sample document fragments from PAN 2015

Based on our observations, it is not unusual for an AV model to identify some salient features in either K or U, U yet fail to find a directly comparable case in the other member of the pair. An example consisting of two brief segments from different authors is shown in Figure 3. We can immediately notice that document u contains unusual words "*maketh*" and "*leadeth*" which are Old English. In contrast, document k is written in relatively colloquial and modern English. A naive method of AV one may devise in this scenario is to detect whether document K contains the usage of "*makes*", the modern counterpart to "*maketh*". If there are occurrences of "*makes*" in K,

we may be able to conclude that the two documents are from different authors. However, the issue with this approach, is the non-zero probability of K containing no usages of "*makes*" at all.

Although it is possible to overcome the problem of non-comparability hand-crafted features, feature engineering is often a labor-intensive process that requires manual labeling. It is also improbable to design all possible features that encode all characteristics of all words. On the other hand, while some modern neural network-based methods were built upon the concept of distributed representations (word embeddings) and could encoded some of the essential features, there is no existing approach explicitly attempting to address the non-comparability problem.

To address the non-compatibility problem, we formulate the NWS, which can be seen as a universal way to distinguish between a pair of documents and solve the AV task in most scenarios in an unsupervised manner. The differences or similarities between documents are determined with respect to the NWS. To this end, we establish a new metric called Deviation Vector Distance (DV-Distance). To the best of our knowledge, the proposed approach is the first model designed with non-compatibility in mind from the ground up.

2.2 Related Works in Authorship Verification

Much of the existing works in AV are based on vocabulary distributions, such as n-gram frequency. The hypothesis behind these models is that the relative frequencies of words or word combinations can be used to profile the author's writing style [64, 18]. One can conclude that two documents are more likely to be from the same author when the distributions of the vocabularies are similar. For example, in one document, we may find that the author frequently uses "*I like ...*," while in another document, the author usually writes "*I enjoy ...*". Such a difference may probably indicate that the documents are from different authors. This well-studied approach has had many successes, such as settling the "Federalist Papers' dispute" [60]. However, its results are often less than ideal when dealing with a limited data challenge.

The number of documents in K and U is often insufficient to build two comparable uni-gram word distributions, let alone 3-gram or 4-gram ones. The depth of difference between two sets of

documents is often measured using the unmasking technique while ignoring the negative examples [32]. This one-class technique achieves high accuracy for 21 considerably large (over 500K) e-Books. A simple feed-forward three-layer auto-encoder (AE) can be used for AV, considering it a one-class classification problem [45]. Authors observe the behavior of the AE for documents by different authors and build a classifier for each author. The idea originates from one of the first applications of AEs for novelty detection in classification problems [25].

AV was studied to detect linguistic traits of sock-puppets to verify the authorship of a pair of accounts in online discussion communities [35]. Recently, a spy induction method was proposed to leverage the test data during the training step under the "out-of-training" setting, where the author in question is from a closed set of candidates while appearing unknown to the verifier [19].

In a more realistic case, we have no specified writing samples of a questioned author, and there is no closed candidate set of authors. Since 2013, a surge of interest arose for this type of AV problem. [61] investigated whether one document is one of the outliers in a corpus by generalizing the Many-Candidate method by [33]. The best PAN 2014E method optimizes a decision tree. Its method is enriched by adopting a variety of features and similarity measures [15]. For PAN 2014N, the best results are achieved by using fuzzy C-means clustering [52]. In an alternative approach, [34] generated a set of impostor documents and applied iterative feature randomization to compute the similarity distance between pairs of documents. One of the more exciting and powerful approaches investigates the language model of all authors using a shared recurrent layer and builds a classifier for each author [1]. Parallel recurrent neural network and transformation AE approaches produce excellent results for various AV problems [20], ranging from PAN to scientific publication's AA [6]. A non-machine learning model comprised of a compression algorithm, a dissimilarity method, and a threshold was proposed for AV tasks, achieving first place in two of four challenges [16].

2.3 Normal Writing Style and Deviation Vector

To make a small and often cross-domain document pair comparable, we propose comparing both documents to the NWS instead of directly comparing the pair. We can define the Normal Writing

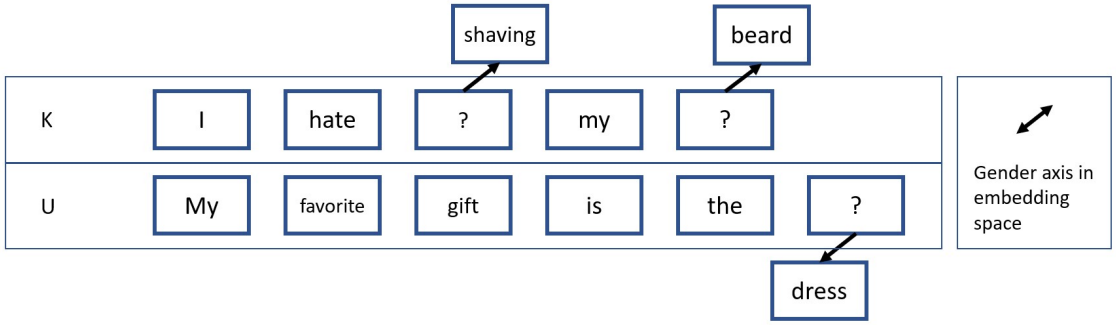


Figure 4: A conceptual demonstration of deviation vector pointing to opposite direction.

Style or NWS loosely as what average writers would write on average, given a specific writing genre, era, and language. From a statistical perspective, the NWS can be modeled as the averaged probability distribution of vocabulary at a location, given its context. As manifested in Figure 3, the reason words *"maketh"* and *"leadeth"* stand out in the document u because they are rarely used in today's writing. Hence, they are *deviant* from the NWS.

We hypothesize that we can utilize modern neural language models to the model NWS. We also assume the predicted word embedding at a given location is an excellent semantic proxy of what an average writer would write at that location. Moreover, we also hypothesize that, generally, an author has a consistent deviation direction in the word embedding space. Consequently, if two documents k and u have the same direction of deviation, then the two documents are likely from the same author. Conversely, if two documents have a significantly different direction of deviation, then they are probably from different authors. Previous empirical evidence shows that word embeddings constructed using neural language models are good at capturing syntactic and semantic regularities in language [50, 51, 56]. The vector offsets encode properties of words and relationships between them. A famous example demonstrating these properties is the embedding vector operation: "King - Man + Woman = Queen", which indicates that there is a specific vector offset that encodes the gender difference.

Given the above context, we theorize that it is possible to encode the deviance of *"maketh"* from *"makes"* as "Maketh - Makes" in a similar manner. We shall refer to the offset vector calculated this way as the Deviation Vector (DV). Figure 4 shows an illustrative example that visualizes the

roles of Normal Writing Style modeling and the DVs. In the upper part of the figure, a document k by a male author is suggested, containing a sentence, "I hate shaving my beard." At the bottom half of the figure, we can see a document u written by a female author: "My favorite gift is a dress." Assuming we have an NWS model that can correctly predict all the words except at locations marked using a question mark, the NWS may predict very general terms, such as "do" or "thing" in place of those words. The actual words at these locations deviate from these general terms in the direction of the DV, represented in the figure using arrows. This specific example contains the words "beard" and "dress", usually associated with a particular gender, while the general terms are gender-less. The DV must then have a component along the direction of the gender axis in the embedding space but in the opposite direction.

2.4 Language Model

We used the AWD-LSTM architecture [48], implemented as part of the universal language model (ULMFit) [22], and RoBERTa [40] to model the NWS. AWD-LSTM is a three-layered LSTM-based language model that is trained by predicting the next word given the preceding sequence. Meanwhile, RoBERTa is a BERT-based model trained by predicting the masked word given an input sequence. Both of these language models are pre-trained on large corpuses. Thus, their predicted embedding for the unseen words can be used as a proxy of the statistical distribution of *NWS*.

Assuming these language models can adequately model the *NWS*, the DVs can be calculated by subtracting the actual embeddings of the words from the predicted word embeddings; more formally, for an input sequence consisting of n tokens $\{w_1, ..., w_n\}$. We use EMB to denote the embedding layer of the language models and use LM to denote the language model itself. Then $EMB(w_i)$ and $LM(w_i)$ will correspond to the embedding of the actual token at location i and the predicted embedding by the language model at location i when the corresponding token is the next

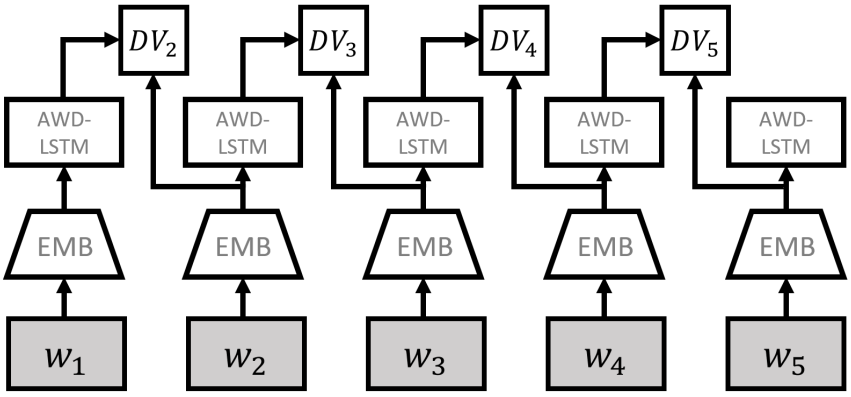


Figure 5: A demonstration of the process of calculating DV using AWD-LSTM.

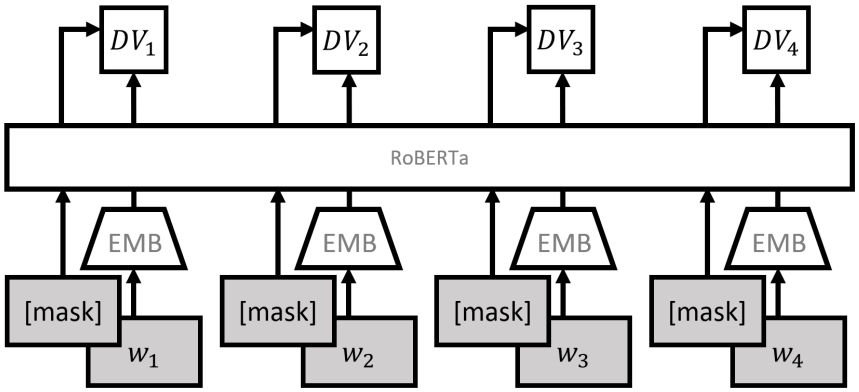


Figure 6: A demonstration of the process of calculating DV using RoBERTa.

token (AWD-LSTM) or is masked (RoBERTa). The DV at location i can then be calculated as:

$$DV_i = LM(w_i) - EMB(w_i) \tag{1}$$

Figures 5 and 6 demonstrate the respective processes of calculating the DVs for a given input sequence using AWD-LSTM and RoBERTa. For AWD-LSTM, at each token location i, the DV is calculated by subtracting the predicted embedding generated at the previous token location $i-1$ by the embedding the current word at i. Consequently, for a document of n words, a total of $n-1$ DVs can be generated. For RoBERTa, the predicted embedding at location i is obtained by feeding the model a complete input sequence with the token at i replaced by the "[mask]" token. A total of n such inference needs to be conducted to obtain all the predicted embeddings at each location.

The DVs can then be calculated by subtracting the predicted embeddings using the actual token embeddings, resulting in a total of n DVs.

2.5 Unsupervised Method: DV-Distance

To compare the direction of a deviation between two documents, we calculate the element-wise mean of all the DVs throughout each document to obtain the "averaged DVs"(ADVs). For a given document of n tokens, $ADV(doc) = \Sigma_{i=1}^{n} DV_i/n$. Notice that for locations with larger deviations between LM and EMB, the corresponding DV shall exert a larger influence on the document level ADV. ADVs are calculated for both K and U, and then the DV-Distance can be calculated as the cosine similarity between $ADV(K)$ and $ADV(U)$.

$$DVDist(K,U) = \frac{ADV(K) \cdot ADV(U)}{\|ADV(K)\| \, \|ADV(U)\|} \tag{2}$$

Since the DV-Distance method is completely unsupervised, the resulting distance values are relative instead of absolute. That is, it is difficult to determine the classification result of a single document pair. Instead, a threshold value needs to be determined such that we can then classify all the document pairs — with DV-Distance values greater than the threshold as "Not same author" and vice versa. To determine the threshold, we follow previous PAN winners, such as [1], and use the median of the DV-distance values between all K and u pairs within the dataset. Using this scheme is reasonable because PAN AV datasets are guaranteed to be balanced.

During our experiments, we found that the threshold value is relatively stable for a particular model in a given dataset, although it, can be quite different between LSTM and BERT models. For real-world applications, the threshold value can be determined ahead of time using a large dataset of the same genre and format as the problem to be evaluated.

2.6 Supervised Method: DV-Projection

One of the significant deficiencies of our DV theory is that it assumes that all differences in the DV hyperspace are relevant. However, one can imagine this assumption does not always hold in all the

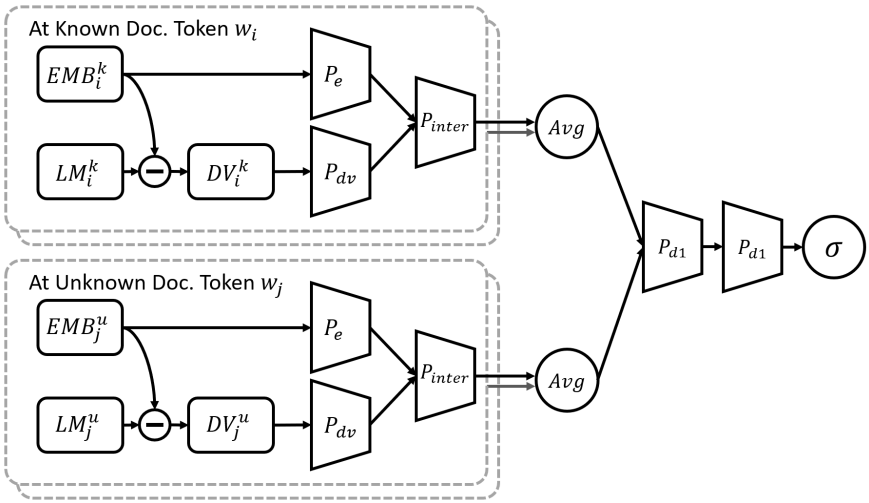


Figure 7: This figure shows the network architecture of the DV-Projection method. Vectors EMB, LM, and DV are represented using a rounded rectangles. Fully connected layers are represented using trapezoids. Element-wise mathmatical operations are represented using circles.

AV settings. For example, the gender dimension shift shown in Figure 4 can be a useful clue when conducting AV on a Twitter dataset or in the context of autobiographies. It may be less relevant if the gender shift occurs in a novel, as the vocabularies used in the novel are more relevant to its characters' genders instead of the author's.

To address this issue, we propose using a supervised neural network architecture to project the D.V.s onto axes that are most helpful for distinguishing authorship features. As we will demonstrate in the results and analysis section of this work, these DV projections are very effective when combined with the original token embeddings generated using the language models.

Here, we shall formally define the DV-Projection process. Given that we have the embeddings and DVs for both a known document and an unknown document, each denoted using EMB_i^k, DV_i^k, EMB_i^u, DV_i^u, we apply dense layers P_e and P_{dv} on embeddings and DVs respectively to extract prominent features. These features are then feed together into the dense layer P_{inter} to allow these vectors to interact with each other. The outputs of P_{inter} are then average-pooled along the sequence to produce document-level features. Lastly, features from both known and unknown documents are connected to two additional fully connected layers P_{d1}, P_{d2} to produce the final output. These operations are summarized in Equation 3 and visualized in Figure 7, and all

layers are used in combination with hyperbolic tangent as the activation function:

$$
\begin{aligned}
TokenFeature_i^k &= P_{inter}(P_e(EMB_i^k), P_{dv}(DV_i^k)) \\
TokenFeature_j^u &= P_{inter}(P_e(EMB_j^u), P_{dv}(DV_j^u)) \\
DocFeature^k &= AvgPool(TokenFeature^k) \\
DocFeature^u &= AvgPool(TokenFeature^u) \\
logit &= P_{d2}(P_{d1}(DocFeature^k, DocFeature^u))
\end{aligned}
\tag{3}
$$

To allow the training of the above model together with RoBERTa, we break documents from the original training document pairs into segments of 128 tokens long. These short document segments are then used to build small training example pairs. This approach allows us to build more training examples to train the network parameters, but also forces the model to be more robust and prevent overfitting by limiting the amount of text it has access to. The training loss used is binary cross entropy loss in combination with the sigmoid function.

Because the DV-Projection method is a supervised model, the model can learn the optimal threshold for classification from a theoretical perspective, therefore eliminating the need for using median value as the threshold. However, the document segment-based training pair generation method can generate significantly more "same author" pairs than "different author" pairs. Therefore the resulting trained model is biased and cannot be assumed to have a 0 valued threshold [1]. To make it consistent, we also use the testing set median value as the threshold for the DV-Projection method [2].

[1] In a real-world application, this problem can be easily addressed by simply generating a large and balanced training dataset.

[2] One can also opt to use the training set median value as the threshold. To give a rough impression of how this will impact the performance: On the PAN14N dataset, using the testing set median value as the threshold will produce 61% in accuracy, using training set median value as the threshold will produce 65% in accuracy. On the PAN14E dataset: using testing set median value as the threshold will produce 73% in accuracy, using training set median value as the threshold will produce 70% in accuracy.

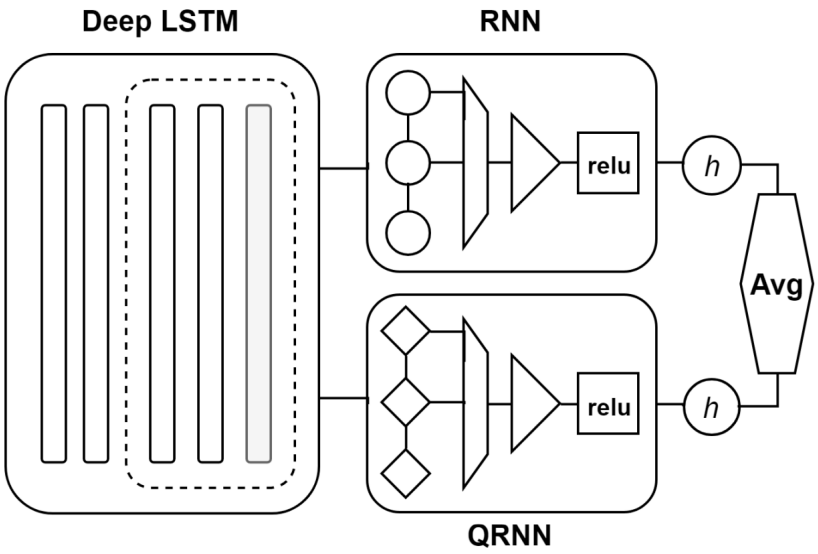


Figure 8: Network architecture of the 2WD-UAV model.

2.7 2WD-UAV

In this section, we will introduce a novel neural network architecture, 2WD-UAV. The model utilizes a pre-trained multi-layer language model in combination with additional RNNs to encode both known documents K and unknown document u. Similar to our DV-Distance methods introduced earlier this chapter, 2WD-UAV also makes use of transfer learning and language modeling, and it demonstrates solid performance in AV problems. On the other hand, from an architectural standpoint, the model resembles many conventional end-to-end trained classification models, in contrast to the unsupervised DV-Distance method. As such, we will use this model as a strong baseline for us to compare and analyze against.

In a gist, the 2WD-UAV model is a bi-directional pipeline of recurrent neural networks (see Figure 8). It is built on top of a pre-trained 5-layer LSTM model, with the last three layers (2 intermediate hidden ones and the final embedding output) acting as inputs by pooling them together. We use an ensemble of sequence classifiers, one based on an RNN and the other using a QRNN [7], a recent addition to the RNN family that combines some properties of recurrent and convolutional networks. Both are 3-layer models with the last two layers averaged and max pooled,

passed through a Rectified Linear Unit (ReLU), and then to the logit units. We output probabilities rather than labels. The predictions made by RNN and QRNN are then averaged.

The attempt to improve generalization through a bi-directional model brings with it two challenges. First, our pre-trained LSTM model is uni-directional. Second, the QRNN design used in this work does not support bi-directional training. We circumvent the problem by tokenizing and numericizing the text data by first training in a regular fashion on a standard pre-trained Wikipedia model, then loading the numericalized tokens backward, using a model trained on Wikipedia backward. At test time, we reversed each document, giving the normal ones to the forward model and the backward ones to the backward model, then averaging the results of the two runs, effectively reaping the benefits of the equivalent use of a bi-directional RNN.

2.8 Experiments

The goal of the empirical study described in the following section is to validate the proposed DV-Distance and DV-Projection method. For this purpose, we use AV datasets released by PAN in 2013 [28], 2014 [66] and 2015 [65].

2.8.1 Datasets

The 2013 version of the PAN dataset consists of 10 training problems and 30 testing problems. PAN 2014 includes two separate datasets, Novels, and Essays. PAN 2014N consists of 100 English novel problems for training and 200 English problems for testing. PAN 2014E consists of 200 English essay problems for training and 200 English essay problems for testing. PAN 2015 is a cross-topic, cross-genre author verification dataset, which means known documents and an unknown document may come from different domains. PAN 2015 contains 100 training problems and 500 testing problems.

2.8.2 Evaluation Metrics

For each PAN dataset, we follow that year's challenge rules. PAN 2013 uses accuracy, Receiver-Operating Characteristic (ROC) and $Score = Accuracy \times ROC$. PAN 2014 introduces the c@1 measure to replace accuracy to potentially reward those contestants who choose not to provide an answer in some circumstances. This metric was proposed in [55], and it is defined as

$$c@1 = (\frac{1}{n}) \times (n_c + (n_u \times \frac{n_c}{n})), \tag{4}$$

Where n_c is the number of problems correctly classified, and n_u is the number of open problems. The Score for PAN 2014 and 2015 is calculated as the product of c@1 and ROC, $c@1 \times ROC$.

Table 1: Authorship Verification results for PAN datasets.

		PAN14E			PAN14N		
Category	Method	c@1	ROC	Score	c@1	ROC	Score
Baseline	GNB	0.675	0.741	0.5	0.56	0.743	0.416
Baseline	LR	0.675	0.728	0.491	0.515	0.604	0.311
Baseline	MLP	0.7	0.768	0.538	0.54	0.782	0.422
PAN	FCMC [52]	0.58	0.602	0.349	0.71	0.711	0.508
PAN	Frery [15]	0.71	0.723	0.513	0.59	0.61	0.36
	TE [20]	0.67	0.675	0.452	0.695	0.7	0.487
	2WD-UAV [5]	**0.73**	0.761	0.555	0.68	**0.801**	0.552
Our model	**DV-Dist. L**	0.58	0.575	0.334	**0.82**	0.79	**0.648**
Our model	**DV-Dist. R**	0.52	0.526	0.274	0.71	0.739	0.525
Our model	**DV-Proj. R**	**0.73**	**0.778**	**0.569**	0.61	0.668	0.41
		PAN13			PAN15		
Category	Method	Acc.	ROC	Score	c@1	ROC	Score
Baseline	GNB	0.633	0.795	0.503	0.552	0.78	0.431
Baseline	LR	0.7	0.781	0.547	0.544	0.796	0.433
Baseline	MLP	0.533	0.5	0.267	0.554	0.687	0.381
PAN	MRNN [1]	-	-	-	**0.76**	0.81	0.61
PAN	Castro [8]	-	-	-	0.69	0.75	0.52
PAN	GenIM [61]	0.8	0.792	0.633	-	-	-
PAN	CNG [24]	-	**0.842**	-	-	-	-
	TE [20]	0.8	0.835	0.668	0.748	0.75	0.561
	2WD-UAV [5]	**0.82**	0.825	**0.677**	0.75	0.822	0.617
Our model	**DV-Dis. L**	0.7	0.763	0.534	**0.76**	**0.834**	**0.634**
Our model	**DV-Dis. R**	0.63	0.746	0.472	0.716	0.767	0.548

2.8.3 Other Baselines

Classic Models with N-gram Features: In our study we use a set of baselines reported in [20]. These results are produced using seven sets of features, including word n-grams, POS n-grams, and character 4-gram. The features need to be transformed because baselines are standard classification algorithms. According to the authors, simple concatenation of two documents' features produces poor results. Seven different functions were used to measure the similarity between feature vectors from both documents, including *Cosine Distance*, *Euclidean Distance*, and *Linear Kernel*. Several common classifiers are trained and evaluated using these similarity measurements, providing a reasonable representation of the performance that is achievable using classic machine learning models and n-gram feature sets. Out of all the baseline results, three classifiers with the highest performance are reported along with the other PAN results for comparison. The selected classifiers are Gaussian Naive Bayes (GNB), Logistic Regression (LR), and Multi-Layer Perceptron (MLP). We compare them with the proposed approach along with the state-of-the-art methods.

PAN Winners: We compare our results to the best performing methods submitted to PAN each year. The evaluation results of the participant teams are compiled in the overview reports of PAN 2013 [28], 2014 [66] and 2015 [65]. In **PAN 2013**, the best-performing methods are the General Imposters Method (GenIM) proposed by [61] and the Common N-Gram (CNG) dissimilarity measure proposed by [24]. In **PAN 2014** challenge, the best method for the English Essay dataset is proposed by [15] (Frery), and the best method for English Novel dataset is by [52] which uses Fuzzy C-Means Clustering (FCMC). In **PAN 2015**, the Multi-headed Recurrent Neural Networks (MRNN) proposed in [1] outperforms the second-best submission (Castro) [8] of the same year by a large margin.

Transformation Encoder: In [20], an auto-encoder based AV model performed competitively on PAN. We include its results to evaluate our model against one of the newest and strongest performers.

2.9 Results and Discussion

Table 1 shows the results from experiments on PAN datasets, detailed in Section 2.8. For each dataset and each evaluation metric, the highest values and the best performing models are marked in bold font. The proposed unsupervised DV-Distance method conducted using AWD-LSTM and RoBERTa is denoted as "DV-Dist. L" and "DV-Dist. R", respectively. The proposed supervised DV-Projection method is trained using DVs produced by RoBERTa and is labeled as "DV-Proj. R" in the table. We were only able to train the projection model on PAN14E and PAN14N because both have relatively long documents and large training sets.

For PAN 2013, our results are slightly below the best performer of that year in terms of accuracy and AUC-ROC; the 0.1 difference in accuracy translates to 3 problems difference out of 30 testing problems. The PAN 2013 corpus are text segments from published Computer Science textbooks. The best performing model in this dataset is the neural network-based model from 2WD-UAV.

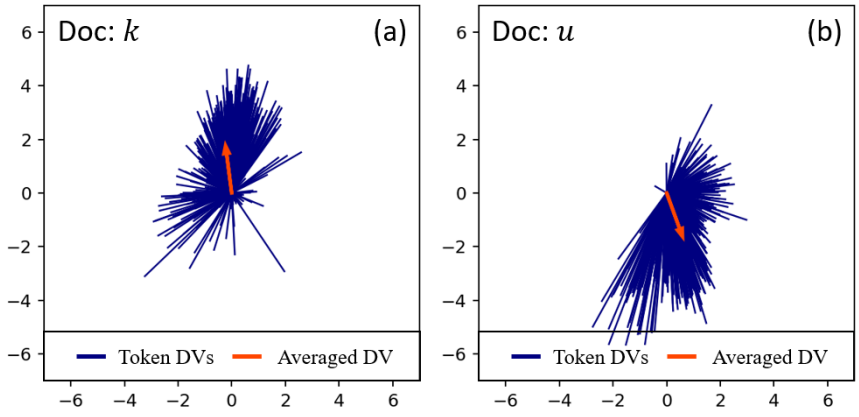
For PAN 2014, we observed some interesting results. For the Novels part of the challenge, our unsupervised DV-Distance method based on LSTMs drastically improves upon previous state-of-the-art models, surpasses the previous best result by 18 percent. On the other hand, for the Essay dataset, both unsupervised DV-Distance methods failed to capture the feature necessary to complete the task, showing only 58% and 52% in accuracy. However, the supervised DV-Projection method successfully projects the DVs generated using RoBERTa into a hyperspace that is suitable for the essay AV problems, resulting in significant performance improvement over the unsupervised models and slightly outperforms the previous best result from 2WD-UAV.

PAN 2015 edition places its focus on cross-genre and cross-topic AV tasks. Based on our observations, the corpus mainly consists of snippets of novels of different genres and sometimes poems. Our proposed DV-Distance method based on multi-layer LSTMs once again shows excellent performance in this dataset, slightly outperforms the previous best model MRNN [1]. In cross-domain settings like PAN 2015, the problem of non-comparability is likely to be very pronounced. Therefore, the strong performance of our methods in this dataset verifies that these methods are quite robust against domain shift and non-comparability.

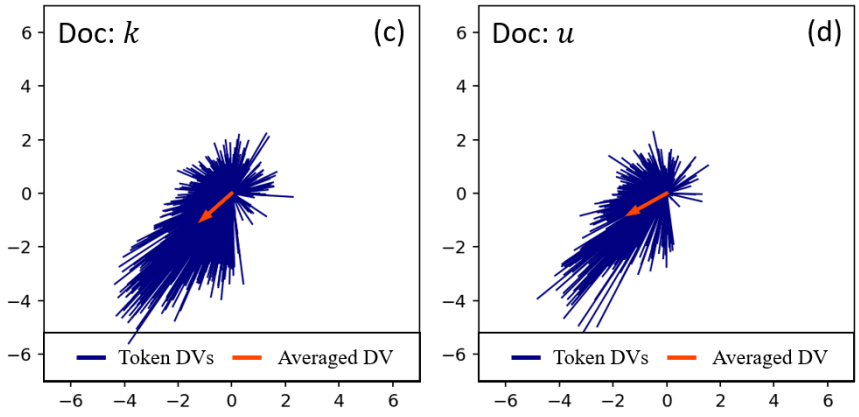
Overall, we have observed two consistent trends in our experiments. First, we find that the AWD-LSTM based DV-Distance method consistently performs better than the RoBERTa based DV-Distance method. At first glance, this may seem counter-intuitive, as BERT-based models are generally regarded as one of the best performing models for language modeling. We theorize that this is precisely the culprit: RoBERTa was able to predict the target word much more accurately, both due to its architectural advantage and it simply has access to more contextual information. However, if the language model is performing "too accurate", it failed to act as a model which represents averaged writing style, but instead mimicking the author's tone and style. From a mathematical perspective, predictions that are "too accurate" will cause DVs calculated using equation (1) to have a magnitude close to zero. Consequently, later steps in equation (2) or (3) will have very little information to work with.

Second, we find that our proposed methods are most suitable for novel and fiction-type documents. Our methods demonstrated state-of-the-art performance in both PAN 2014 Novel and PAN 2015; both consist of mainly novel documents. On the other hand, PAN 2013 and PAN 2014 essay contains writing styles that are more formal and academic-oriented, for which our models performed less competitive. We theorize that this is because essay documents are easier to predict, whereas novels are much more "unpredictable". This difference in predictability means higher quality DVs can be obtained in novel datasets; while in essay datasets, the language models are once again making predictions that are "too accurate", corroborating the first theory we discussed above.

Deviation vectors of two PAN 2015 document pairs are visualized in Figure 9. Figure 9a and Figure 9b shows DVs from two documents written by different authors while Figure 9c and Figure 9d shows DVs from two documents by the same author. The plots are generated by conducting PCA on the DVs at each word, projecting the 400 dimensions DVs from AWD-LSTM to 2 dimensions. A longer line in the plots hence represents a bigger deviation from the NWS. We can observe that in Figure 9a and 9b the DVs' directions are in opposite direction while in Figure 9c and 9d their directions are similar.



DVs of a document pair by different authors.



DVs of a document pair by the same author.

Figure 9: Visualization of deviation vectors in 2D. Each line corresponds to a word-level DV. All DVs in a document are visualized in one subplot. The arrow in each subplot represents the averaged DV direction of that document.

2.10 Conclusion

In this chapter, we presented two novel approaches to the AV problem. Our first method, DV-Distance, relies on using deep neural language models to model the Normal Writing Style and then computes the directional differences in the embedding space between the input document. The other proposed approach, DV-Projection, is a supervised architecture that takes in deviation vectors and extracts relevant features to predict the authorship of the unknown document. The evaluation shows that authorship style differences strongly correlated with the distance metric we

proposed. Our methods demonstrate state-of-the-art models on multiple datasets, both in terms of accuracy and speed.

3 Multi-aspect Sentiment Analysis

3.1 Related Works in Multi-aspect Sentiment Analysis

Neural networks have been shown to be very effective in several natural language processing (NLP) tasks such as semantic parsing [76], sentence modeling [29], and various other classic NLP tasks [10, 69]. Recent works have employed convolutional neural networks (CNN) for sentiment classification at the sentence level and for short texts [14], which have shown promising results. A document-level sentiment analysis system based on recurrent neural network (RNN) has also been explored in [67]. Despite these successes, not many neural-network-based models specifically address the problem of MASA. Although it is always possible to train one network for each predetermined aspect independently, we hypothesize that providing multiple aspect scores concurrently during training should result in better performance. More recently, [59] proposed a hierarchical neural network that has shown promising results in the ABSA tasks. However, they are solving sentence-level ABSA problems, which are different from the document-level ABSA this chapter addresses.

In [72], the multi-aspect rating task was performed using generative modeling. Later, in [73], a unified generative model for rating analysis was proposed that did not require explicit aspect keyword supervision. However, the model does not utilize aspect ratings of a document but instead uses overall ratings to discover latent aspects and estimate ratings on each aspect. A supervised LDA-like [47] scheme was proposed in [70] and later in [41] that regressed the local and global topics (aspects) of reviews with the overall rating and aspect ratings for each review. Ranking algorithms are designed to either identify important aspects [77], or aspect rating prediction without discovering them [62]. Another model used document-level multi-aspect ratings as a form of "weak supervision" to uncover sentence aspects. While it was pretty successful in the sentence aspect attribution task, its primary purpose was not to estimate sentiment ratings [46].

There are many research efforts around SemEval 2015 and 2016 ABSA datasets [57]. In these datasets, both aspect and sentiment polarity labels are available at both sentence and document levels. As such, the works such as [68, 59] address a somewhat different problem than the one in

this work. Both of these works utilize the sentence level labels that are not genuinely available in real-world review datasets and require labor-intensive labeling. Furthermore, the datasets include an extensive set of aspect categories. In contrast, real-world datasets such as the TripAdvisor review dataset have a fixed small set of aspects (roughly 3 - 5) that users rate on at the document level. These small differences in problem setting ultimately lead to very different solutions and models, and we believe both problem settings have their values.

In recent years, deep learning-based models have dramatically changed the field of natural language processing and significantly improved the performance of document classification [75, 6, 79], machine translation [2], and language modeling [13]. CNN [31], RNN such as LSTM/GRU [68, 59] and more recently, pre-trained Bert based architectures have been proposed to solve the problem of sentiment analysis, and these models have significantly advanced state-of-the-art. Pre-trained transformer-alike architectures such as BERT with an extra task-specific layer are fine-tuned on domain reviews for aspect extraction and sentiment classification separately [74]. Unlike the models that extract aspects and predict their sentiments individually, multi-task neural learning frameworks are designed to prevent error propagation in such models. They jointly tackle aspect extraction, and sentiment classification tasks using modern neural networks [71, 17, 38].

3.2 Why SAAM

Most of these previously mentioned deep learning classification models are built on top of some form of *base model* (also called an *encoder*). Most of these base models can take in an embedded sequence of text and generate output vectors at each token's location. These token-level outputs are then fed to one or several layers of fully connected layers (also called *decoders* or *classification heads*) to estimate the probability distribution at the document level ultimately. While much progress has been made in improving these base models' expressiveness, little attention has been paid to the connection between token level outputs of the base models and the final prediction outputs. These token-to-doc connections are often either done by max-pooling/average-pooling [31, 21] or directly use the last/first token's output embedding as the document level embedding [59, 74].

We believe these existing token-to-doc connection schemes are not expressive enough and can become an information bottleneck in both the training and inferencing stage. In comparison, our SAAM framework provides an expressive connection between each sentence and the document-level outputs. In doing so, the SAAM framework can further estimate the latent aspect distribution in each sentence, along with its sentiment rating score. Such fine-grained analysis capability, which we refer to as LSAA, provides more insight into the data. As typical document-level sentiment classification or regression is a unison of sentiments expressed in various sentences across different aspects.

Secondly, our model only requires overall and aspect document-level ratings during the training stage, which can be acquired by most online review systems that use formats similar to those illustrated in Figure 1 and 2. In other words, the proposed SAAM architecture does not require any sentence-level aspect or sentiment supervision and can be easily applied to most existing review datasets and systems.

Last, by assigning each sentence to a proper aspect, the SAAM framework's LSAA capability allows the generation of aspect-specific sentiment snippets. This feature is similar to a summarization system, where the summarization is based on choosing the relevant sentences under different latent aspects. These three major differences not only allow our model to improve upon the current

MASA methods, but also take into account variations of sentiment analysis tasks from different perspectives.

3.3 Sentiment-Aspect Attribution Module

3.3.1 Problem Formulation

Formally, we refer to the text content part of a review simply as *review* in the remaining chapters and denote a single review using r. We use s_i to refer to the ith sentence of a document and a document thus consists of $|s|$ number of sentences. The set of factors that a reviewer can evaluate and rate are referred to as aspects, denoted using A. Moreover, the notation $|A|$ is used for the cardinality of set A. For example, the hotel review data we are working with contains the following aspects:

$$A = \{Value, Room, Location, Cleanliness, Service\}$$

The actual overall rating and aspect ratings associated with a review r are denoted as $R_{overall}(r)$ and $R_{aspects}(r)$. To correspond to the five-star rating scheme, we assume that the overall rating is scalar and that a aspect rating is a vector consisting of $|A|$ number of elements:

$$\begin{aligned} R_{overall}(r) &\in \{1, 2, 3, 4, 5\}^1 \\ R_{aspects}(r) &\in \{1, 2, 3, 4, 5\}^{|A|} \end{aligned} \tag{5}$$

3.3.2 SAAM Classification–1 (SAAM-C1)

The first variation of the SAAM classification model estimates the overall rating class using all features generated from all sentences by the convolution layer or the GRU cell directly. Each sentence's features are also passed into a fully connected softmax layer to estimate the five-class rating distribution of each sentence, correspondingly. There is one such layer for every sentence in an input r while the weights are shared. We refer to these layers as *rating score layers*. Another set

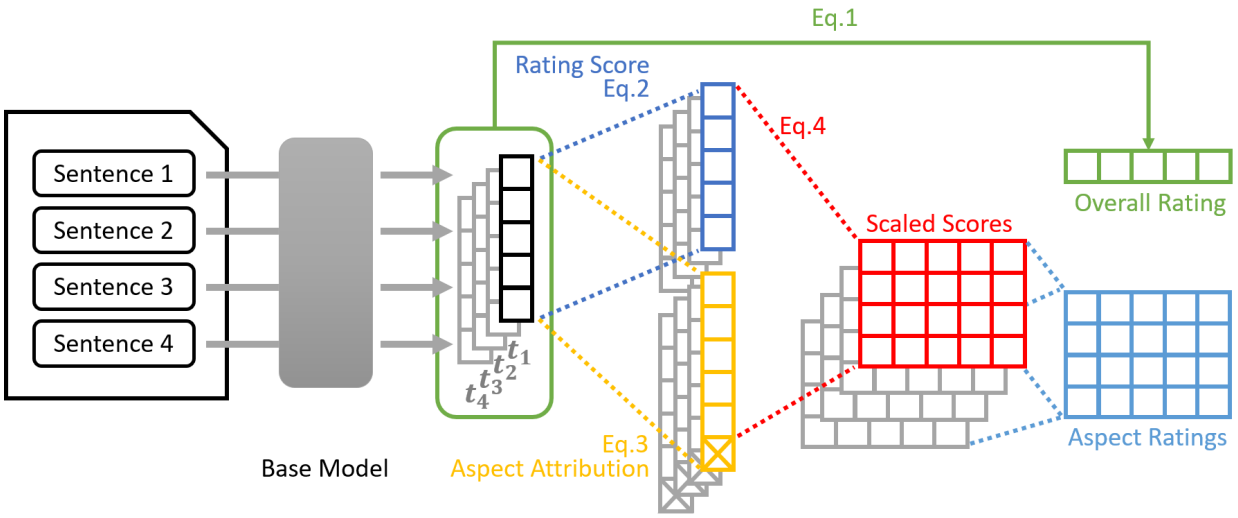


Figure 10: Architecture of SAAM Classification - 1

of weights are used to estimate the aspect distribution of each sentence. We refer to these layers as *aspect attribution layers*. The resulting aspect distributions at the *aspect attribution layers* are then used to scale the rating scores from the *rating score layer* of each sentence, such that sentences with a high probability of belonging to a specific aspect exert a stronger influence on the ultimate aspect rating distributions at the document level. All scaled rating scores are then summed up for each aspect to estimate the final rating class for each aspect. The structure of this SAAM variation, together with the underlying K-CNN base, is visualized in Figure 10.

More formally, given any base model such as a CNN or a GRU and an input document r, we should be able to generate the vector representation $\boldsymbol{t}$ of dimension d for each sentence of the document. The SAAM utilizes these sentence-level feature vectors generated by the base networks to estimate latent distributions and, ultimately, the sentiments of the document. For CNNs, these sentence representations are usually generated using max-pooling the filter activations along the sentence length dimension. For RNNs, this embedding can be obtained by using the final output at the last token. While for BERT-based models, the sentence embedding is usually generated by averaging all outputs along the sequence dimension or using the outputs of the `[CLS]` token. Thus, for an input document with $|s|$ number of sentences, a matrix $\boldsymbol{u}$ of dimension $|s| \times d$ can be obtained. This process is illustrated on the left side of Figure 10.

To obtain a probability distribution of the overall rating label of the the entire review, all of the features in $\boldsymbol{u}$ are then passed into a fully connected softmax layer, the *overall rating layer*. Weights that corresponding to the overall rating are labeled with a superscript o. This operation is shown in Figure 10 in which a green arrow is marked with Equation 6.

$$L_{overall}(r) = softmax(\boldsymbol{W}^o \cdot \boldsymbol{u} + \boldsymbol{b}^o) \tag{6}$$

Like we discussed in the beginning of this section, to estimate the rating distribution of other aspects and carry out LSAA, feature values extracted from each sentence are fed into a *rating score layer* and an *aspect attribution layer*. For sentence s_i, the rating scores (un-normalized distribution) of sentence s_i over $|C|$ rating classes are calculated by the following:

$$score\,(s_i) = (\boldsymbol{W}^a \boldsymbol{t}_i + \boldsymbol{b}^a) \tag{7}$$
$$\text{where} \quad \boldsymbol{W}^a \in \mathbb{R}^{d \times |C|} \quad \text{and} \quad \boldsymbol{b}^a \in \mathbb{R}^{|C|}$$

In the case of a five-star rating scheme, the above $|C|$ would equal 5. This operation is demonstrated in Figure 10, in which the *rating score layer* of each sentence is shown in yellow; four of such layers are drawn.

On the other hand, regarding the *aspect attribution layer*, for a review of total $|A|$ aspects, we actually calculate the aspect attribution for sentence s_i over $|A| + 1$ aspects. The reason for this additional last element in each vector of attribution distribution, which we referred to as *attribution to other-aspect*, is designed to relax the restriction of the model to some extent. It essentially allows the attribution process to ignore the rating scores of some sentences if it deems necessary. Empirically, this structure does make the optimization process faster and allows the

models to provide a better result.

$$aspect(s_i) = softmax(\boldsymbol{W}^r \boldsymbol{t}_i + \boldsymbol{b}^r) \tag{8}$$
$$\text{where} \quad \boldsymbol{W}^r \in \mathbb{R}^{d \times (|A|+1)} \quad \text{and} \quad \boldsymbol{b}^r \in \mathbb{R}^{(|A|+1)}$$

It should be noted in Equation 7 and Equation 8 that, the same $\boldsymbol{W}^a$ and $\boldsymbol{W}^r$ are shared across all sentences. Four *aspect attribution layers* are shown in Figure 10 marked in blue.

Here, computing $aspect\,(s_i)$ should result in a vector $\mathbb{R}^{|A|+1}$ with the first $|A|$ elements representing how strongly the sentence s_i is associated with each aspect. We use $aspect\,(s_i)_{[1:|A|]}$ to denote these first $|A|$ elements. On the other hand, the last element of each aspect attribution, denoted as $aspect\,(s_i)_{[|A|+1]}$, is not associated with any of the actual aspects. We refer to it as the *attribution of other-aspect*. As we will later explain, this additional attribution dimension provides the model with the flexibility to determine if some sentences do not belong to any of the given aspects.

The first $|A|$ elements of the *aspect attribution layer* then distribute output from the *rating score layer* into respective aspects. More specifically, the scaled score for aspect j of sentence s_i would be equivalent to the following:

$$scaledScore\,(s_i)^j = aspect\,(s_i)_{[j]} \cdot score(s_i) \tag{9}$$

In Figure 10 this process is marked in red. It should be noted how $scaledScore\,(s_i)$ is also equivalent to an outer product of the previous two layers, resulting in an matrix of size $\mathbb{R}^{(|A|+1)\times|C|}$, where row j of this matrix is $scaledScore\,(s_i)^j$.

Lastly, these scaled scores for all sentences in a review are summed up element-wise per aspect. A softmax is then applied to the resulting vector to determine the distribution over rating classes

for each aspect of document r:

$$L^{j}_{aspect}(r) = softmax\left(\sum_{i=1}^{|s|} scaledScore(s_i)^j\right) \tag{10}$$

This is shown in the bottom right corner of Figure 10 marked using light blue. The L_{aspect} only contains the rating distribution of aspects — it does not include overall rating distribution. Since rating distribution for overall is directly evaluated using all sentence features $\boldsymbol{u}$ at the *overall rating layer*. Also, it is worth noting that the distribution $L^{|A|+1}_{aspect}(r)$ is not used to estimate any label of the input document; it is the result of *attribution to other-aspect* and hence disregarded.

3.3.3 SAAM Classification–2 (SAAM-C2)

The second variation of the classification model is very similar to the first one. The only difference in this case is that we do not use a separate weight $\boldsymbol{W}^o$ to directly estimate the overall rating distribution. Instead, overall rating is predicted in a similar manner to other aspects, utilizing the sentence aspect attribution process. More specifically, this means for each sentence s_i, an *aspect attribution layer* of size $|A| + 2$ is used, as follows:

$$aspect\,(s_i) = softmax\,(\boldsymbol{W}^r \boldsymbol{t}_i + \boldsymbol{b}^r) \tag{11}$$
$$\text{where} \quad \boldsymbol{W}^r \in \mathbb{R}^{(d)\times(|A|+2)} \quad \text{and} \quad \boldsymbol{b}^r \in \mathbb{R}^{(|A|+2)}$$

Naturally, to estimate the overall rating of a review, we use the $(|A| + 1)$th element of the attribution layer: $aspect\,(s_i)_{[|A|+1]}$ to scale sentence level rating scores towards the overall rating. These scores are then summed together and normalized using a softmax operation similar to SAAM-C1.

The main advantage of this modification over SAAM-C1 is the significant reduction in the size of parameters as the original overall weight matrix $\boldsymbol{W}^0$ is large. Classification-2 can thus use less memory and is potentially less prone to overfitting. Moreover, this scheme can estimate the latent

aspect attribution towards the overall aspect, if such information is indeed a point of interest.

3.3.4 SAAM Regression (SAAM-R)

Apart from the more traditional rating classification task, we also present a variation of the SAAM in which the output layers are changed to real-value regression for rating scores while retaining the sentiment-aspect attribution mechanism. In this setting, the five-star rating distribution is translated to a real value in the range of 1 to 5.

The regression variation of the architecture is architecturally similar to the first version of the classification model. We still connect all the features from all sentences to the output layer for the overall score. However, in this case the overall output of the network is no longer a distribution over the rating classes, but a score without non-linearity. In addition to that, the score is normalized using the sentence count of the corresponding document:

$$L_{overall}(r) = \frac{(\boldsymbol{W}^o \boldsymbol{u} + b^o)}{|s|} \quad \text{where} \quad \boldsymbol{W}^o \in \mathbb{R}^{|s| \times d} \tag{12}$$

Similarly, for each sentence we have a scalar score

$$score\,(s_i) = \boldsymbol{W}^a \boldsymbol{t}_i + b^a \tag{13}$$

On the other hand, the *aspect attribution layer* is kept the same as Classification-1 Equation 8 in this paradigm. The sentence-level scalar score of sentence s_i is then scaled by multiplying with aspect weights. So for aspect j, this is calculated by the following:

$$scaledScore\,(s_i)^j = aspect\,(s_i)_{[j]} \times score\,(s_i) \tag{14}$$

This operation results in a total of $|A|$ scalar score for each sentence, with each value corresponding to one of the aspects. And the final score for aspect j is calculated by the following:

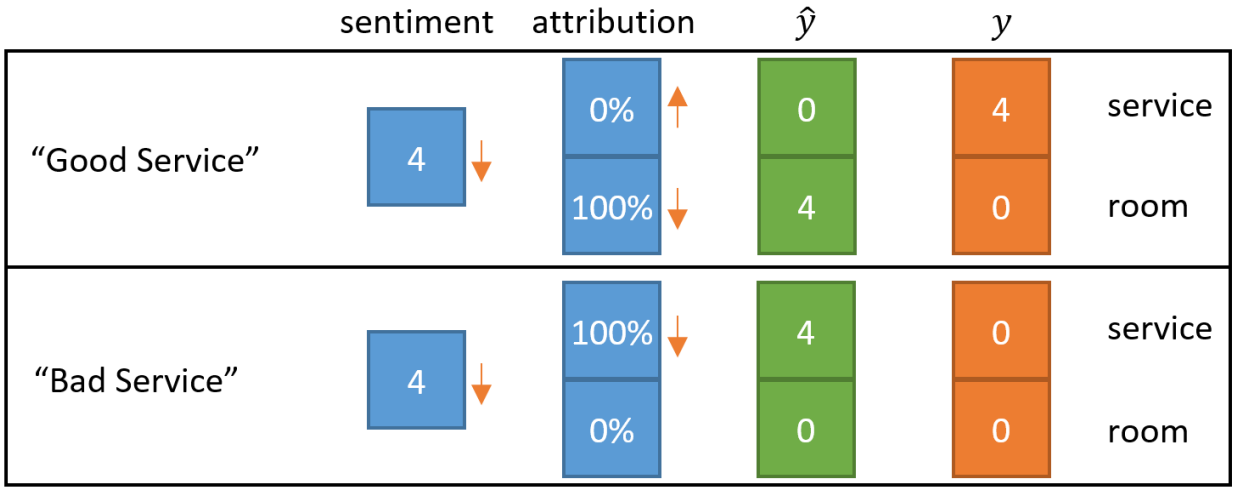


Figure 11: Optimization of attribution layer

$$L^{j}_{aspect}(r) = \frac{\sum_{i=1}^{|s|} scaledScore\,(s_i)^j}{\sum_{i=1}^{|s|} aspect\,(s_i)_{[j]}} \tag{15}$$

Notice the regression scores for aspects are normalized differently compared to the overall score as shown in Equation 12: the scoring for each aspect is normalized with the total probability assigned to that aspect by the attribution layer, instead of the number of sentences in the corresponding review. This normalization makes the aspect scoring process equivalent to a weighted average of sentence aspect scoring, with attribution distribution being the weights.

This difference in normalization is due to the overall score being designed to be an average of sentence scores - it would be problematic if a longer review with a high number of positive sentences goes above the 1-5 score range - assuming the padding sentences getting scores close to 0. On the other hand, the attribution layer has the capability to "discard" scores from the padding sentences when calculating the aspect scores by assigning them a 100% weight on the *attribution of other-aspect*, that is, $aspect\,(s_i)_{[|A|+1]}$. Hence the sentence count normalization is no longer needed.

3.3.5 Intuitions

We provide a simplified example to demonstrate how latent attribution can discover the correct aspect when provided with sufficient examples. Figure 11 shows two training documents, each

containing only one sentence, being processed using a simplified SAAM-R model. We picked the regression model and only two aspects, "Service" and "Room", for easier demonstration. However, the idea should be able to generalize to other variants as well as more aspects.

Blue boxes are internal parameters produced by SAAM's rating scoring layer and Attribution Layer (marked as *sentiment* and *attribution* respectively). Green boxes are output values resulting from the element-wise product of the former two layers. Lastly, orange boxes are ground truth scores.

Recall that for SAAM-Regression, the document-level aspect rating predictions are calculated by the following:

$$\hat{y} = sentiment \otimes attribution \tag{16}$$

and the loss can be expressed simply as: $loss = (\hat{y} - y)^2$.

For the first sentence, "Good Service", let us assume the sentiment layer produces the correct score, but the attribution layer is wrong by attributing all the sentiment scores into room aspect. When optimizing the sentiment layer and attribution layer using gradient descend, the gradients' directions are indicated using orange arrows for each blue value. As can be seen here, the attribution layer will be slightly adjusted towards the correct attribution, which is service 100% and room 0%.

In the second example, "Bad Service", let us assume the attribution layer this time produces the correct aspect distribution, but the sentiment layer mistakenly produced a very high sentiment score. In this case, the gradient descent will pass through the attribution layer and decrease the sentiment score.

Although, in both examples, the gradient descent process has produced some side effects: the sentiment layer in the first case and the attribution layer in the second case were optimized in the wrong direction. However, ultimately, given enough examples and training steps, the system should converge correctly. Imagine a third sentence, "Good Room", with correct ground truth, in which circumstances the optimization process should have only one possible solution in the blue boxes.

3.4 Experiments and Evaluations

3.4.1 Data

We used the TripAdvisor hotel review dataset from [72], and the BeerAdvocate data previously used in works such as [46] to examine the performance of our framework.

The TripAdvisor dataset consists of 108,891 reviews across 1,850 hotels. Each hotel review in the original raw data is associated with one overall rating and five aspect ratings - "Value", "Room", "Location", "Cleanliness", and "Service". For our experiment, only reviews with more than three sentences and all of the five aspects rated were selected. Of the 14,906 reviews that meet the above requirements, 75% and 25% of the documents were selected as training and testing sets, respectively. One thousand reviews were picked from the training set as a development set to tune hyper-parameters. After we determined these hyper-parameters, the models were re-trained using all available training samples.

Similar parsing and selection process were also applied to the BeerAdvocate dataset, and 100,000 beer reviews were selected for our experiment. Aspects associated with each beer review include "Appearance", "Taste", "Palate", and "Aroma". Among these, the aspect "Palate" can be roughly understood as "mouthfeel". The advantage of this dataset is that its aspects are more independent of each other than those in the hotel reviews. For example, "Value", "Room", and "Cleanliness" are often strongly correlated. This property of the BeerAdvocate dataset allowed us to better evaluate the sentence-level aspect attribution process's correctness. As with the TripAdvisor dataset, 75% and 25% of the reviews were selected as the training and testing sets, respectively, and five thousand reviews were selected as a development set for tuning model parameters.

We used the TripAdvisor hotel review dataset to evaluate our classification modules SAAM-C1 and SAAM-C2, and used the BeerAdvocate beer review dataset to evaluate our regression variant of the module SAAM-R.

3.4.2 Evaluation of Document-level MASA

Because our proposed SAAM is an add-on module that can be combined with many different modern neural network architectures, the document-level sentiment analysis performance of the whole model (base + SAAM) was determined using both components. In the following experiments, we opt to use two representative models as base models to better highlight the characteristics of SAAM.

The first base model is the K-CNN, as proposed in [31]. In its original form, this model uses a total of 300 convolutional filters (100 of each size) to extract features from each review. A fully connected softmax function is then applied to estimate the label probability distribution. We trained separate models for each aspect of the reviews as baselines. We then replaced the fully connected layers at the end of the K-CNN with our SAAM to demonstrate how our method improved the performance of the overall model.

Furthermore, we have also included a version of CNN, which we refer to as Expanded CNN (E-CNN), to demonstrate that the performance improvement we observed from using SAAM was not merely due to an increase in the number of parameters. Specifically, in this baseline, reviews are also divided into sentences. Each sentence is then passed to the CNN layer to generate 300 dimension embedding. All of the features generated from sentences are then concatenated and passed to a fully connected softmax layer for classification. Notice that, this formulation mimics how the overall rating is estimated in the SAAM-C1 scheme (3.3.2) we proposed, shown in Equation 6. A total of $|A|$ such fully connected softmax layers in the model are used to concurrently train and estimate all aspects.

The second base model we chose is a GRU-based RNN [9]. Similar to the CNN-based model, we set the hidden state vector to 300 and trained a separate model for each aspect of the dataset as baselines. We then replace the final fully connected layers with our SAAM to demonstrate its flexibility and performance improvement over the base models.

We have also included three classification baselines: Hierarchical LSTM [59], Doc2Vec [36] and SVM [27] for TripAdvisor hotel review classification task, and two regression baselines: Linear

Table 2: Performance of proposed SAAM classification variants against corresponding base models and other baselines, experimented on TripAdvisor hotel review dataset.

	Overall		Aspect 1 Value		Aspect 2 Room		Aspect 3 Location		Aspect 4 Cleanliness		Aspect 5 Service		Avg	Avg
	Acc	MSE	Acc	MSE	Acc	MSE	Acc	MSE	Acc	MSE	Acc	MSE	Acc.	MSE
K-CNN	58.0	0.715	50.8	0.943	45.1	1.061	44.8	1.302	47.5	0.995	50.3	1.319	47.70	1.124
E-CNN	58.6	0.600	49.9	0.883	41.8	1.135	42.9	1.107	46.1	1.076	48.6	1.224	45.86	1.085
CNN+SAAM-C1	58.3	0.706	**51.6**	0.888	**47.2**	**0.985**	44.7	1.308	**50.2**	1.042	**51.6**	**1.138**	**49.06**	1.072
CNN+SAAM-C2	58.0	0.62	**51.8**	**0.803**	**48.2**	**0.906**	**45.3**	1.166	**49.3**	**0.927**	**51.0**	**1.039**	**49.12**	**0.968**
RNN	58.2	0.647	51.4	0.891	44.9	1.158	43.5	1.467	45.9	1.214	48.4	1.209	48.72	1.098
RNN+SAAM-C1	56.6	0.722	**54.9**	**0.772**	**49.0**	**0.976**	**45.8**	1.407	**49.8**	**1.041**	**51.5**	**1.100**	**51.27**	1.003
RNN+SAAM-C2	**60.2**	**0.625**	**54.1**	**0.824**	**49.5**	**0.969**	**46.6**	**1.279**	**50.4**	**1.021**	**52.3**	**1.052**	**52.19**	**0.962**
Hi-LSTM	61.6	0.533	54.7	0.751	46.4	1.029	44.8	1.216	47.1	1.052	48.7	1.234	50.5	0.969
Doc2Vec	54.1	0.829	47.8	1.087	42.3	1.305	44.7	1.439	45.1	1.291	47.3	1.585	45.44	1.341
SVM	29.2	1.892	35.5	2.368	33.9	2.368	8.4	9.010	32.5	1.917	33.3	2.375	28.72	3.608

Regression and SVM regression [27] for the BeerAdvocate beer review regression task. We included these referencing baselines to help readers interpret the difficulty of our task and dataset and use them as a benchmark for estimating the expressiveness of our proposed modules. The Hierarchical LSTM proposed in [59] consists of two levels of LSTM networks: one working at the word level to generate sentence embedding vectors, and another that takes these sentence embedding vectors as input, and estimates sentiment polarity for each sentence in a document. To adapt this model as one of our baselines, we modified the model by concatenating the last output vectors from the sentence-level bi-directional LSTM and feeding the resulting vector through several dense layers, where each layer corresponds to one of the aspects.

It should be noted that the additional computational time required to train the SAAM is not significant, as the additional parameter matrices W^a and W^r are relatively small. We tested all SAAM variants on one Nvidia Titan RTX GPU; the increase in training time was around 10% to 30% longer than that of the base model CNN and RNN, respectively.

3.4.3 MASA Results

Table 2 presents the results of classification variants SAAM-C1, SAAM-C2 on top of CNN and GRU-RNN compared to the base version of those two models, evaluated on the TripAdvisor testing set. Both prediction accuracy (Acc) and mean squared error (MSE) were calculated for overall and each of the five aspects. The predicted rating classes were assumed to be real values when calculating

Table 3: Performance of proposed SAAM regression variants against corresponding base models and other baselines, experimented on BeerAdvocate beer review dataset.

	Overall		Aspect 1 Appearance		Aspect 2 Taste		Aspect 3 Palate		Aspect 4 Aroma		Average	
	MSE	R2	MSE	R2	MSE	R2	MSE	R2	MSE	R2	MSE	R2
E-CNN	0.267	0.423	0.228	0.325	0.260	0.454	0.239	0.425	0.258	0.408	0.246	0.403
CNN+SAAM-R	0.264	0.431	**0.208**	**0.386**	**0.207**	**0.564**	0.220	0.471	**0.219**	**0.498**	**0.213**	**0.480**
RNN	0.256	0.448	0.209	0.383	0.231	0.514	0.237	0.429	0.243	0.445	0.235	0.443
RNN+SAAM-R	**0.228**	**0.508**	0.195	0.424	**0.182**	**0.617**	**0.202**	**0.514**	**0.199**	**0.542**	**0.201**	**0.521**
Linear Regr	0.307	0.338	0.255	0.246	0.266	0.440	0.287	0.308	0.285	0.346	0.273	0.335
SVM	0.272	0.414	0.226	0.332	0.235	0.505	0.253	0.391	0.252	0.421	0.242	0.412

MSE. In the right-most two columns, the average accuracy and MSE for the five aspects were shown for easier comparison. In Table 2, bold text highlights statistically significant performance improvement of SAAM applied models compared to their corresponding base models.

The Table 2 indicate that our proposed SAAM-C1 and SAAM-C2 models provide a consistent performance improvement over their corresponding base models. More specifically, stacking SAAM-C1 and SAAM-C2 on top of a CNN and an RNN improves the aspect sentiment classification accuracy by an average of 2 to 3 percent. In certain aspects such as *Room* and *Cleanliness*, the improvements in accuracy are as much as 5 percent. We also note that there is little to no improvement in the *Overall* rating classification. One reason for this could be that overall sentiment classification is relatively easy, as the model does not need to learn aspect-specific feature combinations, and reviewer behavior is more consistent for overall ratings.

As a reference, Hi-LSTM provided an additional 1 to 4 percent improvement in accuracy when compared with the base version of RNN. This improvement is likely due to the additional expressiveness offered by the second layer of LSTM, which can selectively pass through sentence-level features to document level output to allow more accurate distribution estimation. In other words, the performance advantage of Hi-LSTM can be attributed to its more expressive sentence-to-document connection. As a comparison, after combining SAAM-C1 and C2 with the RNN base model, the performance gaps between the RNN and Hi-LSTM were eliminated and, in some cases, reversed, indicating that SAAM significantly improves the expressiveness and information flow from sentence level to document level.

Table 3 shows the performance of our SAAM regression models (SAAM-R) based on a CNN and an RNN, as well as base models and other referencing baselines evaluated using the BeerAdvocate beer review data testing set. Bold text highlights statistically significant performance improvements of SAAM applied models compared to their corresponding base models. Once again, we can see that by adding our SAAM regression module to the base model, we can significantly reduce the error when comparing against base models. Among all aspects, we observed that base models CNN and RNN have relatively poor performance on aspect *Taste* and *Aroma*, which may have occurred because the language used to describe these two aspects is very similar. The attribution mechanism in our model can alleviate this issue by redirecting the latent sentence-level sentiment to the correct aspect.

3.4.4 Evaluation of Latent Sentence-level Aspect Attribution

One of our SAAM framework's key advantages is that it can leverage the latent aspect attributed to each sentence and organically combine them. In this section, we evaluate the LSAA facet of our models. Two human labelers manually labeled 1,000 sentences with aspects in each of the datasets. For both of the datasets, the set of possible labels included names of the existing aspects and an additional label, "*None*," which indicates the labeler did not believe the sentence was related to any of the aspects. The labeling from two labelers achieved a Cohen's Kappa agreement score of 0.66 on the hotel dataset, indicating significant, but not perfect, agreement. On the other hand, the beer review dataset yielded a better agreement score of 0.70, reinforcing our observation that aspects in beer reviews are more independent and unambiguous.

In addition to the human-generated labels, we made use of the review format many reviewers followed in the BeerAdvocate dataset as an additional set of ground truth. More specifically, many reviewers on BeerAdvocate use "A:", "S:", "M:" and "T:" to signify the beginning of corresponding review segments[3]. We selected around 16,000 sentences that have these prefixes and marked them with the corresponding correct labels.

[3] "A" for "Appearance"; "S" for "smell", corresponding to the "Aroma" aspect; "M" for "mouthfeel", corresponding to the "Palate" aspect; "T" for "Taste".

Table 4: Evaluation of our SAAM framework's ability to estimate latent sentence aspects. Accuracy is reported against labels generated independently by two humans on both datasets and a keyword-based labeling method of the BeerAdvocate dataset.

	Hotel 1	Hotel 2	Beer 1	Beer 2	Beer Keywd
CNN+C1	0.32	0.35	-	-	-
CNN+C2	0.48	0.47	-	-	-
CNN+R	-	-	0.63	0.61	0.87
GRU+C1	0.46	0.50	-	-	-
GRU+C2	0.55	0.52	-	-	-
GRU+R	-	-	0.68	0.64	0.95

To obtain the sentence-level latent aspects determined by SAAM, we examined the estimated latent aspect distribution ($aspect\,(s_i)$) for that sentence. If the dominant value of the learned aspect distribution was consistent with the human labeler's aspect, it is considered as a correct attribution. Table 4 shows these evaluation results of SAAM-C1 and SAAM-C2 on the hotel review dataset, and of SAAM-R on the beer review dataset. We can observe that almost all model combinations can attribute sentences to aspects with reasonably high accuracy. Among these, the regression model based on GRU had the highest performance in this task. Moreover, we can see that the regression models yielded even stronger agreement with the keyword-based labeling. This indicates that the models successfully learned these keywords and used them as strong signals when conducting latent aspect attribution.

It is worth noting that due to inherent overlapping between aspect categories, reviewer subjectivity, and vague nature of some of the aspects, this LSAA task is non-trivial. Considering it is a latent variable and there are 4 to 5 potential classes, the above results indicate good performance.

3.4.5 Snippet Extraction

In addition to estimating latent aspect distribution, SAAM can also estimate the latent sentiment distribution ($score\,(s_i)$) for each sentence. We believe there is much exciting opportunity for information extraction by combining this latent information discovered through LSAA. This section demonstrates one interesting possible application of aspect-specific review snippet extraction, that

is inspired by several existing review summarization work such as [37]. Particularly, it is interesting for cases in which the overall review rating is positive but one aspect is evaluated negatively (or vice-versa); our model may be able to explain such discrepancies. Here, we will present some qualitative results by SAAM-R to provide an intuitive understanding of this application and the SAAM framework.

Review 1, 5 Stars Overall: "*spent 5 days at excellence at Punta Cana, most of the people who work at the hotel were very pleasant ...*"
Sentiment snippet for the Service aspect via the lowest sentiment score:

- *"internet service was not available in the room and barely in the lobby area"* [**Service, -2.89**]

Review 2, 1 Star Overall: "*I do not know where to start. the roaches in the room, the rude waiters, bartenders, front desk, the dead flies that stayed on our friends' mirror the entire stay, the average at best food ...*"
Sentiment snippet for the Location and Cleanliness aspects via the highest sentence score:

- *"the beach was fabulous"* [**Location, 5.99**]
- *"the resort itself, décor, pool, beach access was great"* [**Cleanliness, 5.90**]

Review 3, 5 stars Overall with 3 Stars for the "Location" aspect: "*Was awesome. my wife and I traveled to excellence 11/20-11/26 and had a great time ...*"
Sentiment snippet for the Location aspect via the lowest sentence score:

- *"the worst part about this resort is the drive there and back, the roads are terrible and it is over an hour"* [**Location, -1.53**]

Review 4, 4 stars Overall with 2.5 Stars for the "Palate" aspect: "*A: Pours a clear yellow with a mild white head, good retention ...*"
Sentiment snippet by extracting the only Palate sentence:

- *"M: Very light-bodied, watery, light base beer for sure."* [**Palate, -0.96**]

3.5 Conclusion

In this chapter, we presented a novel add-on framework called the sentiment-aspect attribution module (SAAM) that can be combined with common deep learning architectures to solve the problem of multi-aspect sentiment analysis. The proposed SAAM addresses the token-to-doc connection bottleneck problem using an intuitive and expressive latent sentiment-aspect attribution (LSAA) process. Furthermore, the LSAA process also facilitates fine-grained sentiment analysis and summarization. Two classification variants and one regression variant of the SAAM were demonstrated and tested on both CNN- and RNN-based networks. Experimental results on a real-world hotel review dataset and a beer review dataset indicated that our proposed framework yields significant performance improvements over the base networks. Lastly, we also demonstrated the potential of using sentence-level latent features generated by the SAAM for aspect-specific or sentiment-specific snippet extraction. This iteration of the SAAM has several areas of potential improvement. However, we believe this work presents a fascinating new angle to solve multi-labeled document-classification problems.

4 More on Multi-aspect Sentiment Analysis

4.1 Introduction and Motivation

In the last chapter, we introduced a framework called the sentiment-aspect attribution module (SAAM), which can be combined with traditional neural network architectures such as CNNs and RNNs to solve multi-aspect sentiment analysis problems. In addition to document-level classification capability, SAAM also demonstrated that it is possible to discover sentence-level aspect and sentiment information without the need for fine-grained labeling during training. In fact, Only the original user-generated document-level ratings are required for the proposed system to estimates sentence-level sentiment and aspect distributions.

However, with the latest architectural advancements in neural network, many aspects of the original design of SAAM now require revision and improvements. More specifically, three main issues have been identified in the original SAAM model; these issues are described individually below.

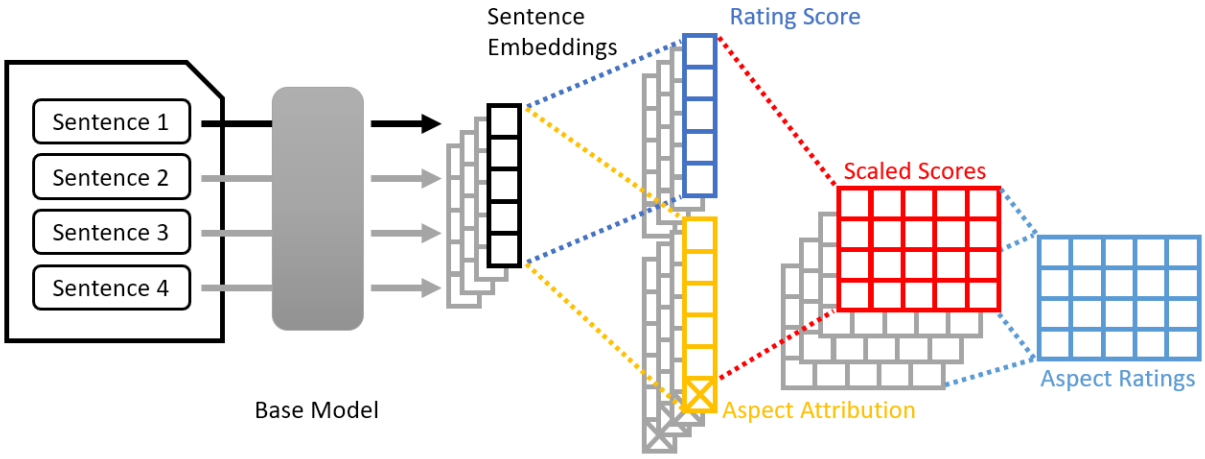


Figure 12: Architecture of the previous version of SAAM framework. The sentence embeddings generated by the base model and the layers for rating and aspect estimations are labeled with corresponding colors.

First, an "information bottleneck" in the original SAAM architecture design severely limits its expressiveness. This bottleneck exists between the sentence-level embedding generated by the base

model and the document-level sentiment distributions. This can be observed in Figure 12, between sentence embeddings (indicated in black) and rating or aspect distributions (indicated in blue and yellow). In most base model settings, the size of sentence embedding vectors d is significantly larger than the number of aspects $|A|$ or the number of rating classes $|C|$. For example, for many modern LSTM-based models, the sentence embedding size d is around 300 to 400 elements. In contrast, the number of aspects $|A|$ and the number of rating classes $|C|$ for a dataset are only around 2 to 6. Consequently, in SAAM, there is a very considerable constraint on how much information each sentence can contribute to the overall document level outputs.

Another perspective of this information bottleneck issue is to compare it against designs of attention mechanism. In some ways, the SAAM framework we proposed can be thought of as an aspect-driven attention mechanism. For regular attention mechanisms, such as [2, 42], the attention alignment score α allows token-level embeddings to interact with each other before they are fed into the decoder model. However, in the SAAM framework, the interaction is conducted at the aspect rating level instead of at the embedding level, so the interaction is greatly limited.

Secondly, the previous SAAM architecture does not provide the base models with context information regarding surrounding sentences in the document. This can be an undesirable property, as it limits the ability of the base model to generate accurate sentence embeddings. For example, in the BeerAdvocate dataset, we often observe that the model is having difficulty figuring out if a particular sentence is discussing the taste of the beer or its smell, as the set of adjectives used in both aspects have a lot in common. However, if some contextual information is provided, such as the previous and subsequent sentences, the model may be better able to attribute the sentence to the correct aspect, thereby resulting in much better document-level estimations.

Lastly, during our experiments with the SAAM module, we found that the complete model can be challenging to train. In some conditions, the trained model will opt to attribute all of the sentences to one single aspect, so other aspects have limited or no sentiment-rating attributions, yet the model still performs reasonably well on the training set. We believe this phenomenon is mostly caused by SAAM overfitting the training set. In this work, we propose to conduct transfer

learning utilizing pre-trained multi-layer language models to alleviate the issue of overfitting.

To address all of the above issues, we propose a neural network module architecture that improves the original SAAM architecture, which is referred to as SAAM v2. As detailed in the following section, almost all aspects of the original SAAM have been altered to improve performance. However, the primary design objectives here remain changed: SAAM v2 is still designed as an add-on module that can be combined with modern neural network architectures (base model) to allow them to perform multi-aspect sentiment analysis (MASA). As demonstrated in the Results section, the models with SAAM v2 show superior performance compared to models with fully connected layers. Furthermore, as indicated in the analysis section, when combining SAAM v2 with a pre-trained multi-layer LSTM base model, our module can estimate latent sentence-level aspects with exceptionally high accuracy.

The remaining sections of this chapter are organized as follows. Section 4.2 details the architecture of the proposed SAAM v2 module. In this section, we will also provide the rationales behind some of the design choices we made to address the three issues introduced above. Section 4.3 discusses our experiment settings and in Section 4.4 details and analysis corresponding experiment results. Lastly, in Section 4.5 we present a few documents with sentence-level outputs generated by SAAM v2.

4.2 Model Architecture

This chapter formally defines our SAAM v2 framework. As previously discussed, SAAM v2 can be combined with common modern neural network architectures and takes their token-level vector outputs as inputs. SAAM v2 redirects and combines information within these token-level vector representations to estimate sentiment for each aspect at the document level. Figure 13 shows the abstracted architecture of our proposed module when combined with a base model. For simplicity, the figure showcased an input document with three sentences, and the model estimates sentiment distribution for two aspects.

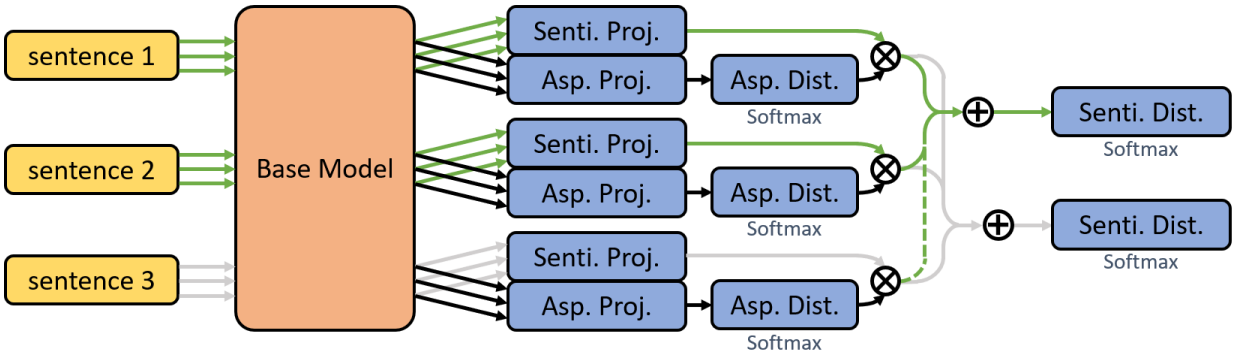


Figure 13: This figure shows the overall architecture of SAAM v2 combined with a base model. The figure illustrates the scenario of an input document with three sentences. The model estimates sentiments over two different aspects.

4.2.1 Encoding Stage

For a given review document $\boldsymbol{r}$ consists of n number of tokens, a base model M can be used to map $\boldsymbol{r}$ into a matrix $\boldsymbol{t} \in \mathbb{R}^{n \times d}$, where d is the base model's output embedding dimension. Almost all modern NLP neural networks produce some form of matrix $\boldsymbol{t}$, examples include CNN based models (K-CNN [31]), RNN based models (AWD-LSTM [48]) and more recently Bert based models (BERT [13], Longformer[3]).

4.2.2 Sentence Feature Extractors

Two separate feature extractors are used to project token-level features to form sentence-level embeddings. After multiple iterations of experiments, we have found that a linear layer with GeLU activation strikes the optimal balance of performance and complexity. More specifically, sentiment and aspect extractors take token-level outputs $\boldsymbol{t}$ as input. Each extractor consists of a linear layer with output dimension d_{senti} and d_{aspt}, respectively. After performing the linear projection and activation, average pooling was then used to combine outputs corresponding to tokens from the same sentence. For example, if input document $\boldsymbol{r}$ consists of $|s|$ number of sentences, then after

applying both extractors to $\boldsymbol{t}$ should result in two matrices of $|s|$ rows.

$$\boldsymbol{s}_{senti} = Extractor(\boldsymbol{t}, W_{senti})$$

$$\boldsymbol{s}_{aspt} = Extractor(\boldsymbol{t}, W_{aspt})$$

Where $\boldsymbol{s}_{senti} \in \mathbb{R}^{|s| \times d_{senti}}$ and $\boldsymbol{s}_{aspt} \in \mathbb{R}^{|s| \times d_{aspt}}$

4.2.3 Aspect Driven Attention

Given the aspect features $\boldsymbol{s}_{aspt}$, a linear layer with a softmax activation function is used to estimate the aspect distribution associated with each sentence. To describe this in another way, we estimate the relatedness of the ith sentence s^i by feeding $\boldsymbol{s}^i_{aspt}$ through a linear layer with a softmax function. Following the paradigm previously applied in SAAM, we increment the output dimension of this layer by one to allow the "attribution to other-aspect". That is, assuming the dataset contains a total of $|A|$ number of aspects, this layer will contain $|A| + 1$ number of output units, where the last output probability corresponds to the "other aspect".

$$AD(s^i) = softmax(\boldsymbol{s}^i_{aspt} W_{AD} + b_{AD}) \tag{17}$$

We then use the estimated aspect distribution of each sentence to attribute the corresponding sentiment embeddings to each aspect. To do so, an outer product between the sentiment feature vector and aspect distribution is performed $\boldsymbol{s}^i_{senti} \otimes AD(s^i)$. As a result, sentiment embedding of each sentence is mapped to a matrix of dimension $d_{senti} \times |A| + 1$. Each row of this matrix is thus the sentiment embedding vector scaled by the aspect distribution, thereby controlling the amount of sentiment contributing toward each aspect at the document level.

These matrices resulting from the outer product from each sentence are then summed element-wise, then normalized by the total weight assigned to each aspect. The resulting document-level matrix will retain the same dimension as the sentence-level ones, with each row of the matrix

represents the sentiment embedding towards a particular aspect. In other words, the sentiment embedding vector towards the a-th aspect, denoted as $m^{[a]}$ can be calculated as follows:

$$m^{[a]} = \frac{\sum_{i=1}^{|s|} \left(\boldsymbol{s}_{senti} \cdot AD(s^i)^{[a]} \right)}{\sum_{i=1}^{|s|} AD(s^i)^{[a]}}$$

4.2.4 Sentiment Estimation

Lastly, the document-level aspect sentiment can be estimated using multiple linear layers with softmax activation. For the j-th aspect, a linear layer takes $m^{[j]}$ as input and outputs a distribution over the possible ratings. We use $|A|$ such layers independently; each corresponds to one of the aspects.

4.3 Evaluation

4.3.1 Evaluation Method

Evaluation of SAAM v2 framework is conducted following a similar strategy as the SAAM from last chapter. Two different datasets were used: the TripAdvisor's hotel review dataset and the BeerAdvocate's beer review dataset. The hotel dataset and beer dataset contain 14,906 and 100,000 reviews, respectively. Both datasets were divided into training and testing set using 75% and 25% split, and 20% of the training set was used as a development for hyper-parameter tuning.

the pre-trained AWD-LSTM network from the fast.ai library and the Longformer network from the Hugging-Face Transformers library were fine-tuned on the training sets and evaluated on the testing sets as baselines. For all baseline models, we connect the base models with a ReLU activated dense layer and then $|A|$ independent softmax dense layers, each of which predicted the sentiment of an aspect. We then replaced the dense layers in baselines with the proposed SAAM v2 module to evaluate the performance improvement over the baseline models. For all the models, we have reported accuracies over all aspects for both datasets.

Table 5: This table shows document-level classification accuracies on each aspect of the BeerAdvocate dataset the TripAdvisor dataset. The performance of both base models and the performance after they are combined with the proposed SAAM v2 are reported.

TripAdvisor Aspects	Overall	Value	Room	Location	Clean	Service	Averaged
AWD-LSTM	66.27	58.49	53.28	47.98	53.89	54.99	55.82
AWD-LSTM+SAAM v2	67.77	60.44	55.58	48.57	54.91	57.40	57.44
longformer	66.97	59.64	53.36	45.81	53.33	53.81	55.49
longformer+SAAM v2	65.71	59.21	54.64	47.55	54.40	56.59	56.35
BeerAdvocate Aspects	Overall	Appearance	Taste	Palate	Aroma		Averaged
AWD-LSTM	63.46	60.44	67.69	63.31	63.19		63.99
AWD-LSTM+SAAM v2	63.84	62.34	68.23	63.50	64.08		64.39
longformer	62.79	60.74	67.39	63.24	62.88		63.41
longformer+SAAM v2	63.76	62.26	68.24	63.88	64.08		64.44

4.3.2 Training Details

We applied LayerNorm and DropOut before each linear layer to regularize the SAAM V2 module, including layers inside sentence feature extractors, aspect-driven attention mechanism, and the sentiment prediction layers.

Adam optimizer was used with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e-6$ for gradient descent during training . For the learning rate scheduler, we used cosine-annealing with roughly 50,000 cycles equivalent warm-up period. The maximum learning rate was set to $2e-5$. Early-stopping was used for Longformer-based models by monitoring the accuracy on validation set to avoid over-fitting.

4.4 Results

The evaluation results for the BeerAdvocate review dataset and the TripAdvisor review dataset are presented in the top and bottom half of Table 5, respectively. The sentiment classification accuracies of the base models AWD-LSTM and Longformer are reported on rows marked with "AWD-LSTM" and "Longformer". The performances of our proposed SAAM v2, when combined with these base models, are reported on the rows beneath them. The right-most column of each row shows an averaged result of all accuracy values reported on that row for easier comparison.

Overall, these experimental results have demonstrated that, by stacking SAAM v2 on top of

these state-of-the-art base models, we are able to further improve their performance by around 1-2 percent. These improvements are consistent across all aspects in both the hotel review dataset and the significantly larger beer review dataset.

One notable observation is that base model AWD-LSTM is out-performing Longformer in most of the tasks evaluated here, despite the Longformer is generally considered the superior model with significantly more parameters. We believe this is because the document classification tasks at hand are relatively easy and hence cannot fully utilize the additional expressiveness afforded by the attention mechanisms driving the modern Bert-based architecture. Nonetheless, the proposed SAAM v2 module focuses on addressing the information flow from token-level embeddings to document-level predictions. Consequently, we can observe that in both datasets, the models with SAAM v2 improved the the base models and yielded better performance.

4.5 Analysis of Sentence-level Attribution

One useful property of the SAAM v2 model is its ability to estimate sentence-level aspect distribution without requiring any sentence-level ground-truth. For each individual sentence of the document, the corresponding aspect distribution estimated by SAAM v2 can be obtained according to Equation 17. More formally, $AD(s^i)$ can be viewed as the estimated aspect distribution for sentence s^i over $|A|$ aspects.

For evaluation, we manually labeled 1000 sentences for both the TripAdvisor hotel review dataset and the BeerAdvocate beer review dataset. We then computed top-1 accuracy and top-2 accuracy for sentence-level aspect attribution by comparing human generated ground-truths with the estimated distributions. Here, top-1 accuracy is defined as the percentage of labeled sentences for which the aspect with the highest probability matches the ground-truth. Whereas, top-2 accuracy is defined as the percentage of sentences for which the aspect with the highest of second-highest probability matches the aspect labeled as the ground-truth.

For the TripAdvisor dataset, the accuracy of **AWD-LSTM+SAAM v2**'s sentence-level aspect attribution is as follows:

- definitely not a 5 star resort i 'm dumbfounded that this hotel gets good reviews and is so highly rated `[overall, value]`
- it 's decidedly a 3 star property , not 5 stars as indicated `[overall, room]`
- the rooms are very dated and run down , old crappy beds and pillows , an old tv and overall poorly maintained `[room]`
- the whole property is pretty run down and old - looking `[room]`
- the food is subpar , not one meal i had would be called great `[overall, service]`
- the service is uneven and the staff is poorly trained and uninformed `[service]`
- many do not comprehend english `[service]`
- the beach is great , it 's the only redeeming factor `[location]`
- however the resort is a 1- hour taxi trip from the airport `[location]`

Figure 14: A sample hotel review with aspects estimated to sentences by our SAAM v2 in brackets.

- Top-1 Accuracy: 73.2
- Top-2 Accuracy: 88.6

For the BeerAdvocate dataset, the accuracy of **AWD-LSTM+SAAM v2**'s sentence-level aspect attribution is as follows:

- Top-1 Accuracy: 89.5
- Top-2 Accuracy: 93.4

To put these numbers into perspective: for the TripAdvisor dataset which contains five aspects, a model that randomly guessed could achieve an accuracy of 20 percent. The previous version of SAAM showed a performance of 55 percent accuracy at the highest. Similarly, for the BeerAdvocate dataset which has four aspects, a model that randomly guessed could achieve an accuracy of 25 percent, whereas the previous best SAAM model reported an accuracy of around 68 percent. Therefore, the new evaluation results for sentence-level aspect attribution represent a massive increase in the model's capability. Furthermore, top-2 accuracies reported here indicate that when SAAM v2's first guess was incorrect, the second guess was almost always right.

- after the one hour ride from the airport we arrived at the hotel and were greeted by everyone we met `[service]`
- i have to say that the staff at the hotel were very nice and made every effort to learn our names and greet us by name each time they saw us `[service]`
- the hotel itself was clean , the staff was very friendly , and nothing ever felt crowded `[service]`
- however , the food was not great `[service]`
- it was not bad - but it was not great `[overall, service]`
- i 'm not a big eater but i was prepared to indulge on my vacation and there just was not anything i was crazy about `[overall, service]`
- we went on two excursions - swimming with the sting - rays/sharks and the zip - line tour `[overall, location]`
- we loved the zip - line excursion `[location]`
- the staff was great and our bus driver and tour guide were great `[service]`
- it was interesting to visit the sting - rays and swim with the sharks but the reef where we snorkeled was disappointing `[location]`
- the fish were very small and there was not much to see `[location]`
- the electricity went out in our room a handful of times , especially when i used the hairdryer `[overall, room]`
- also , our ac was terrible `[room]`

Figure 15: A sample hotel review with aspects estimated to sentences by our SAAM v2 in brackets.

Bibliography

[1] Bagnall, D. Author identification using multi-headed recurrent neural networks. *CEUR Workshop Proceedings 1391* (jun 2015).

[2] Bahdanau, D., Cho, K. H., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015* (sep 2015), International Conference on Learning Representations, ICLR.

[3] Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The Long-Document Transformer. *arXiv* (apr 2020).

[4] Bevendorff, J., Ghanem, B., Giachanou, A., Kestemont, M., Manjavacas, E., Markov, I., Mayerl, M., Potthast, M., Rangel, F., Rosso, P., Specht, G., Stamatatos, E., Stein, B., Wiegmann, M., and Zangerle, E. Overview of pan 2020: Authorship verification, celebrity profiling, profiling fake news spreaders on twitter, and style change detection. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 12260 LNCS* (2020), 372–383.

[5] Boumber, D., Zhang, Y., Hosseinia, M., Mukherjee, A., and Vilalta, R. Robust authorship verification with transfer learning. EasyChair Preprint no. 865, 2019.

[6] Boumber, D., Zhang, Y., and Mukherjee, A. Experiments with convolutional neural networks for multi-label authorship attribution. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC* (2018), European Language Resources Association (ELRA).

[7] Bradbury, J., Merity, S., Xiong, C., and Socher, R. Quasi-recurrent neural networks. In *5th International Conference on Learning Representations, ICLR* (nov 2017), International Conference on Learning Representations, ICLR.

[8] Castro, D., Adame, Y., Pelaez, M., and Muñoz, R. Authorship verification, combining linguistic features and different similarity functions. *Conference and Labs of the Evaluation Forum, CLEF* (2015).

[9] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Doha, Qatar, Oct. 2014), Association for Computational Linguistics, pp. 1724–1734.

[10] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. Natural language processing (almost) from scratch. *Journal of Machine Learning Research 12*, Aug (2011), 2493–2537.

[11] Cortes, C., and Vapnik, V. Support-vector networks. *Machine Learning 20*, 3 (sep 1995), 273–297.

[12] Dave, K., Lawrence, S., and Pennock, D. M. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th International Conference on World Wide Web, WWW* (2003), pp. 519–528.

[13] Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference* (oct 2019), vol. 1, Association for Computational Linguistics (ACL), pp. 4171–4186.

[14] Dos Santos, C., and Gatti, M. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers* (2014), pp. 69–78.

[15] Fréry, J., Largeron, C., and Juganaru-Mathieu, M. UJM at CLEF in author verification based on optimized classification trees: Notebook for PAN at CLEF 2014. Tech. rep., 2014.

[16] Halvani, O., Winter, C., and Graner, L. On the usefulness of compression models for authorship verification. In *Proceedings of the 12th International Conference on Availability, Reliability and Security* (2017), ACM, p. 54.

[17] He, R., Lee, W. S., Ng, H. T., and Dahlmeier, D. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (Florence, Italy, July 2019), Association for Computational Linguistics, pp. 504–515.

[18] Hoover, D. L. Statistical stylistics and authorship attribution: an empirical investigation. *Literary and Linguistic Computing 16*, 4 (nov 2001), 421–444.

[19] Hosseinia, M., and Mukherjee, A. Detecting sockpuppets in deceptive opinion spam. *Computing Research Repository abs/1703.03149* (2017).

[20] Hosseinia, M., and Mukherjee, A. Experiments with neural networks for small and large scale authorship verification. In *International Conference on Computational Linguistics and Intelligent Text Processing, CICLing* (mar 2018).

[21] Howard, J., and Ruder, S. Fine-tuned language models for text classification. *CoRR abs/1801.06146* (2018).

[22] Howard, J., and Ruder, S. Universal language model fine-tuning for text classification. In *Proceedings of the Conference 56th Annual Meeting of the Association for Computational Linguistics, ACL* (jan 2018), vol. 1, pp. 328–339.

[23] Hu, M., and Liu, B. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining* (2004), ACM, pp. 168–177.

[24] Jankowska, M., Kešelj, V., and Milios, E. Proximity based one-class classification with Common N-Gram dissimilarity for authorship verification task. Tech. rep., 2013.

[25] Japkowicz, N., Myers, C., Gluck, M., et al. A novelty detection approach to classification. In *IJCAI* (1995), vol. 1, pp. 518–523.

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

[27] Joachims, T. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, Cambridge, MA, 1999, ch. 11, pp. 169–184.

[28] Juola, P., and Stamatatos, E. Overview of the author identification task at PAN 2013. In *Conference and Labs of the Evaluation Forum, CLEF* (2013).

[29] Kalchbrenner, N., Grefenstette, E., and Blunsom, P. A convolutional neural network for modelling sentences. In *52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference* (2014), vol. 1, pp. 655–665.

[30] Kestemont, M., Tschuggnall, M., Stamatatos, E., Daelemans, W., Specht, G., Stein, B., and Potthast, M. Overview of the author identification task at PAN-2018: Cross-domain authorship attribution and style change detection. In *CEUR Workshop Proceedings* (2018), vol. 2125.

[31] Kim, Y. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[32] Koppel, M., and Schler, J. Authorship verification as a one-class classification problem. In *Proceedings of the Twenty-First International Conference on Machine learning* (2004), Association for Computing Machinery, p. 62.

[33] Koppel, M., Schler, J., and Argamon, S. Authorship attribution in the wild. *Language Resources and Evaluation 45*, 1 (2011), 83–94.

[34] Koppel, M., and Winter, Y. Determining if two documents are written by the same author. 178–187.

[35] Kumar, S., Cheng, J., Leskovec, J., and Subrahmanian, V. An army of me: Sockpuppets in online discussion communities. In *Proceedings of the 26th International Conference on World Wide Web* (2017), International World Wide Web Conferences Steering Committee, pp. 857–866.

[36] Le, Q. V., and Mikolov, T. Distributed representations of sentences and documents, 2014. cite arxiv:1405.4053.

[37] Li, F., Han, C., Huang, M., Zhu, X., Xia, Y.-J., Zhang, S., and Yu, H. Structure-aware review mining and summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)* (Beijing, China, Aug. 2010), Coling 2010 Organizing Committee, pp. 653–661.

[38] Li, X., Bing, L., Li, P., and Lam, W. A unified model for opinion target extraction and target sentiment prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2019), vol. 33, pp. 6714–6721.

[39] Lin, C., and He, Y. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM conference on Information and knowledge management* (2009), ACM, pp. 375–384.

[40] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* (7 2019).

[41] Lu, B., Ott, M., Cardie, C., and Tsou, B. K. Multi-aspect sentiment analysis with topic models. In *2011 IEEE 11th international conference on data mining workshops* (2011), IEEE, pp. 81–88.

[42] Luong, M. T., Pham, H., and Manning, C. D. Effective approaches to attention-based neural machine translation. In *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing* (aug 2015), Association for Computational Linguistics (ACL), pp. 1412–1421.

[43] Luyckx, K., and Daelemans, W. Authorship attribution and verification with many authors and limited data. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)* (2008), Manchester, pp. 335–336.

[44] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies* (Portland, Oregon, USA, June 2011), Association for Computational Linguistics, pp. 142–150.

[45] Manevitz, L., and Yousef, M. One-class document classification via neural networks. In *Neurocomputing* (2007), vol. 70, Elsevier, pp. 1466–1481.

[46] McAuley, J., Leskovec, J., and Jurafsky, D. Learning attitudes and attributes from multi-aspect reviews. In *2012 IEEE 12th International Conference on Data Mining* (2012), IEEE, pp. 1020–1025.

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

[48] Merity, S., Keskar, N. S., and Socher, R. Regularizing and optimizing LSTM language models. In *The International Conference on Learning Representations, ICLR* (2018).

[49] Microsoft Research. Turing-nlg: A 17-billion-parameter language model by microsoft (https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/), 2020.

[50] Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *Computing Research Repository abs/1301.3781* (2013).

[51] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (2013), pp. 3111–3119.

[52] Modaresi, P., and Gross, P. A language independent author verifier using fuzzy c-means clustering: Notebook for PAN at CLEF 2014. In *CEUR Workshop Proceedings* (2014), vol. 1180, pp. 1084–1091.

[53] Mukherjee, A., and Liu, B. Aspect extraction through semi-supervised modeling. In *Proceedings of the 50th annual meeting of the association for computational linguistics: Long papers-volume 1* (2012), Association for Computational Linguistics, pp. 339–348.

[54] Pang, B., and Lee, L. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval 2*, 1-2 (oct 2008), 1–135.

[55] Peñas, A., and Rodrigo, A. A simple measure to assess non-response. *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies 1*, June (2011), 1415–1424.

[56] Pennington, J., Socher, R., and Manning, C. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (2014), pp. 1532–1543.

[57] Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S., AL-Smadi, M., Al-Ayyoub, M., Zhao, Y., Qin, B., De Clercq, O., Hoste, V., Apidianaki, M., Tannier, X., Loukachevitch, N., Kotelnikov, E., Bel, N., Jiménez-Zafra, S. M., and Eryiğit, G. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)* (Stroudsburg, PA, USA, 2016), Association for Computational Linguistics, pp. 19–30.

[58] Popescu, A.-M., and Etzioni, O. Extracting product features and opinions from reviews. In *Natural language processing and text mining*. Springer, 2007, pp. 9–28.

[59] Ruder, S., Ghaffari, P., and Breslin, J. G. A hierarchical model of reviews for aspect-based sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (Stroudsburg, PA, USA, 2016), Association for Computational Linguistics, pp. 999–1005.

[60] Savoy, J. The Federalist Papers revisited: A collaborative attribution scheme. *Proceedings of the American Society for Information Science and Technology 50*, 1 (2013), 1–8.

[61] Seidman, S. Authorship verification using the impostors method. In *CEUR Workshop Proceedings* (2013), vol. 1179.

[62] Snyder, B., and Barzilay, R. Multiple aspect ranking using the good grief algorithm. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference* (2007), pp. 300–307.

[63] Somasundaran, S., Namata, G., Wiebe, J., and Getoor, L. Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing Volume 1 - EMNLP '09* (Morristown, NJ, USA, 2009), vol. 1, Association for Computational Linguistics, p. 170.

[64] Stamatatos, E. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology 60*, 3 (mar 2009), 538–556.

[65] Stamatatos, E., Daelemans, W., Verhoeven, B., Juola, P., López-López, A., Potthast, M., Stein, B., Cappellato, L., Ferro, N., Jones, G., and San Juan, E. Overview of the Author Identification Task at PAN 2015. In *CEUR Workshop Proceedings* (2015), vol. 1391, pp. 877–897.

[66] Stamatatos, E., Daelemans, W., Verhoeven, B., Potthast, M., Stein, B., Juola, P., Sanchez-Perez, M. A., and Barrón-Cedeño, A. Overview of the author identification task at PAN 2014. In *CEUR Workshop Proceedings* (Sheffield, UK, 2014), vol. 1180, pp. 877–897.

[67] Taboada, M., Brooke, J., Tofiloski, M., Voll, K., and Stede, M. Lexicon-based methods for sentiment analysis. *Computational linguistics 37*, 2 (2011), 267–307.

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

[69] Tang, D., Qin, B., and Liu, T. Document modeling with gated recurrent neural network for sentiment classification. In *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing* (Stroudsburg, PA, USA, 2015), Association for Computational Linguistics, pp. 1422–1432.

[70] Titov, I., and McDonald, R. A joint model of text and aspect ratings for sentiment summarization. In *proceedings of ACL-08: HLT* (2008), pp. 308–316.

[71] Wang, F., Lan, M., and Wang, W. Towards a one-stop solution to both aspect extraction and sentiment analysis tasks with neural multi-task learning. In *2018 International Joint Conference on Neural Networks (IJCNN)* (July 2018), pp. 1–8.

[72] Wang, H., Lu, Y., and Zhai, C. Latent aspect rating analysis on review text data: a rating regression approach. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), ACm, pp. 783–792.

[73] Wang, H., Lu, Y., and Zhai, C. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (2011), pp. 618–626.

[74] Xu, H., Liu, B., Shu, L., and Yu, P. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 2324–2335.

[75] Yang, F., Mukherjee, A., and Zhang, Y. Leveraging multiple domains for sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (Osaka, Japan, Dec. 2016), The COLING 2016 Organizing Committee, pp. 2978–2988.

[76] Yih, W.-t., He, X., and Meek, C. Semantic parsing for single-relation question answering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (2014), pp. 643–648.

[77] Yu, J., Zha, Z.-J., Wang, M., and Chua, T.-S. Aspect ranking: identifying important product aspects from online consumer reviews. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1* (2011), Association for Computational Linguistics, pp. 1496–1505.

[78] Zhang, X., Zhao, J., and Lecun, Y. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems* (sep 2015), vol. 2015-January, Neural information processing systems foundation, pp. 649–657.

[79] Zhang, Y., Yang, F., Zhang, Y., Dragut, E., and Mukherjee, A. Birds of a feather flock together: Satirical news detection via language model differentiation, 2020.

[80] Zhao, W. X., Jiang, J., Yan, H., and Li, X. Jointly modeling aspects and opinions with a maxent-lda hybrid. In *Proceedings of the 2010 conference on empirical methods in natural language processing* (2010), Association for Computational Linguistics, pp. 56–65.

[81] Zhuang, L., Jing, F., and Zhu, X.-Y. Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on Information and knowledge management* (2006), pp. 43–50.