Dyadic Memory Networks for Aspect-based Sentiment Analysis

Yi Tay*
Nanyang Technological University
ytay017@e.ntu.edu.sg

Luu Anh Tuan*
Institute for Infocomm Research
at.luu@i2r.a-star.edu.sg

Siu Cheung Hui
Nanyang Technological University
asschui@ntu.edu.sg

ABSTRACT

This paper proposes *Dyadic Memory Networks* (DyMemNN), a novel extension of end-to-end memory networks (memNN) for aspect-based sentiment analysis (ABSA). Originally designed for question answering tasks, memNN operates via a memory selection operation in which relevant memory pieces are adaptively selected based on the input query. In the problem of ABSA, this is analogous to aspects and documents in which the relationship between each word in the document is compared with the aspect vector. In the standard memory networks, simple dot products or feed forward neural networks are used to model the relationship between aspect and words which lacks representation learning capability. As such, our dyadic memory networks ameliorates this weakness by enabling rich dyadic interactions between aspect and word embeddings by integrating either parameterized neural tensor compositions or holographic compositions into the memory selection operation. To this end, we propose two variations of our dyadic memory networks, namely the Tensor DyMemNN and Holo DyMemNN. Overall, our two models are end-to-end neural architectures that enable rich dyadic interaction between aspect and document which intuitively leads to better performance. Via extensive experiments, we show that our proposed models achieve the state-of-the-art performance and outperform many neural architectures across six benchmark datasets.

KEYWORDS

Deep Learning, Neural Networks, Sentiment Analysis

1 INTRODUCTION

Aspect-level Sentiment Analysis (ABSA) can be considered as a fine-grained extension of traditional opinion mining or sentiment analysis tasks. Unlike document level opinion mining which predicts a single polarity for the entire document, this task is concerned with detecting polarities with respect to given entities or aspects. Consider the following example text on a review website, *'Loved the salmon but the waiter was very rude. 2 stars!!'*. Clearly, we understand that this review refers to two aspects, i.e., service and food, in which the polarities for them are opposite (positive for food and negative for service). Hence, the task of aspect-level opinion mining is concerned with providing polarity detection at a more fine-grained level which is intuitively more suitable for practical business and social applications. It is also good to note that the task of ASBA can also be known as *stance detection* [14] and *attitude identification* [11] which are also long standing problems in NLP research.

Deep learning, without a doubt, has demonstrated incredibly competitive performance in many NLP tasks. With no exception, the task of sentiment analysis is also dominated by neural architectures [3, 12, 28, 32]. Apart from the clear demonstration of competitive performance, deep learning removes the need for laborious feature engineering in which cumbersome feature extraction code has to be written for each feature. Amongst the myriad of neural architectures proposed, the long short-term memory network [6] remains a powerful choice as a neural encoder while end-to-end memory networks [11, 27, 30] are the present state-of-the-art for aspect-based sentiment analysis [30].

Memory networks (MemNN), originally incepted for the purpose of question answering, consider an aspect (such as 'service' or 'food') as a query and dynamically select relevant information from the input document. Memory networks are reminiscent of attention mechanisms [13, 17, 24, 33, 36] which have been a wildly fashionable research area of interest in the recent years. Intuitively, memory networks are neural architectures that aim to adaptively select words in the documents that are important to the aspect. Generally, this process is operated via standard similarity measures such as simple dot products [27] or feed forward neural networks [30] which may lack representation learning capability. For example, simple dot products only consider element-wise interaction between vector pairs and even feed forward neural networks do not consider the dyadic interactions between vectors.

As such, in this paper, we propose **two** novel variations of the memory network architecture. We call this class of neural architectures *Dyadic Memory Networks* (DyMemNN) which incorporate rich dyadic interactions between query (aspect) and document into the memory network architecture. The key motivation is that modeling richer interactions between aspect and document will improve the performance in the ABSA task. After all, the memory selection operation lives at the heart of the memory network architecture and is absolutely vital for good performance. , To this end, we exploit two vector composition techniques that enable rich representation learning. Inspired by many recent advances in question-answering [22] and knowledge base completion [25], our first proposed model adopts parameterized neural tensor compositions to drive the memory selection operation. We refer to this model as *Tensor DyMemNN* due to the usage of tensors in the core memory selection operation of the standard memory network. Neural tensor compositions model multiple views of dyadic interactions between two vectors which fit extremely elegantly within the memNN framework. Using tensor compositions to model the relationship between aspect

*Denotes equal contribution.

CIKM'17 , Singapore, Singapore

978-1-4503-4918-5/17/11…$15.00
DOI: 10.1145/3132847.3132936

and words in the document enables richer interaction and should improve the performance of memNN.

Additionally, we also propose a memory efficient adaptation in which we adopt *compressed tensor products*. Our second model is highly related to holographic reduced representations [5, 18] and exploits complex-valued computation, i.e., circular convolutions and circular correlations via Fast Fourier transforms (FFT). This variation boasts of significantly less parameters compared to neural tensor compositions while enabling rich representation learning. Due to its connections with holographic reduced representations, we refer to this model as *Holo DyMemNN* (short for Holographic DyMemNN). Moreover, unlike neural tensor composition which is strictly a symmetric operation, our holographic composition simultaneously exploits **both** symmetric and asymmetric dyadic interactions which intuitively should lead to richer representations.

Finally, the prime contributions of this paper can be summarized as follows:

- We propose *Dyadic Memory Networks* (DyMemNN) for aspect-based sentiment analysis. DyMemNN incorporates composition techniques that model the rich dyadic interactions between aspect and words in a document. To this end, we propose two variations of DyMemNN, namely Tensor DyMemNN and Holo DyMemNN. To the best of our knowledge, we are the first work that incorporates or extends memory networks with neural tensor compositions or holographic compositions.
- Both variations of DyMemNN achieved the state-of-the-art performance on **six** benchmark datasets. Moreover, DyMemNN consistently outperforms MemNN on all settings. Moreover, we obtain performance gains over the standard MemNN in which performance gains can be up to 7% in terms of F1-score and on average $\approx$ 2% improvement. In addition, we also conduct extensive ablation studies to study the effect of the key hyperparameters of our models. We also provide an informed study-driven recommendation on how to use our proposed models and showcase some qualitative examples of the benefits of our proposed models.

2 RELATED WORK

The ability to automatically detect the polarity, attitude and stance of a document is extremely desirable for many practical applications. The wide applicability of sentiment analysis in a diverse range of domains such as finance, politics and businesses has naturally spurred on the rich advancements in sentiment analysis research across both academic and industrial NLP communities. Intuitively, this task is simple, i.e., given a document, determine whether it is positive, negative or neutral. In short, this task can be interpreted as a multi-class (or binary) classification problem. Additionally, our work is concerned with a more fine-grained adaptation of traditional document level sentiment analysis, i.e., we are interested in aspect-level polarity. In this task of aspect-based sentiment analysis, the task is now to determine the polarity of the document with respect to a given aspect.

Across the rich history of sentiment analysis research, many researchers and communities have devoted themselves to feature engineering approaches. In these applications, features such as sentiment lexicons [7, 23], ngram features or parse-tree features [9, 10] are constructed and subsequently, a final feature vector is aggregated and passed through a traditional machine learning classifier such as Support Vector Machines (SVM). While these traditional feature engineering methods remain competitive, they are plagued with flaws which include the cumbersome nature of feature extraction and pipeline based systems that are difficult to maintain in production.

Today, neural architectures are incredibly fashionable for many (if not all) NLP tasks. End-to-end training allows neural models to be built without requiring any feature engineering. This is not only attractive from practical perspectives but also extremely competitive. Li et al. [11] showed that a SVM model trained on an extensive list of handcrafted features could not outperform even simple deep learning architectures such as simple long short-term memory (LSTM) networks [6]. On the other hand, end-to-end memory networks (memNNs) [11, 27, 30] are the present state-of-the-art in this task.

Notably, the key difference in aspect-level sentiment analysis over document-level analysis is the additional requirement and complexity of modeling the relationship between document and aspect. Intuitively, the neural network architecture should adaptively return a different output based on a different aspect. A myriad of neural architectures have been proposed to accomplish this. This includes the Target-Dependent LSTM (TD-LSTM) [29] which trains a bidirectional LSTM network model towards the aspect target. Additionally, ATAE-LSTM and AT-LSTM [33] are recently incepted attentional models which were inspired by recent advancements in natural language inference [24]. AT-LSTM learns to attend by incorporating the aspect vector into the attention mechanism while ATAE-LSTM concatenates aspect vectors right before the LSTM modeling layer.

Our work is mostly concerned with memory networks in which our main contribution resides upon. Memory networks operate via a memory selection operation which is reminiscent of neural attention mechanisms. The key intuition of memory networks is given as follows: Firstly, the semantic similarity between each word in the document is modeled with respect to the aspect. Secondly, this information is used to adaptively select the relevant pieces of information (or memory slices) from the input document. Memory networks are highly competitive neural architectures for many NLP tasks which include question answering [2], logical reasoning and visual question answering [34]. In our domain, Tang et al. [30] first proposed deep memory network architectures for ABSA problems and demonstrated highly competitive results. On the other hand, Li et al. proposed *AttNet* [11], a multi-task adaptation of memNN that learns to jointly predict polarities and the presence of targets.

In our proposed dyadic memory networks (DyMemNN), we adopt two rich compositional techniques for modeling the dyadic interactions between aspect and documents. The first of our two models, the Tensor DyMemNN was inspired by many recent advances in related fields such as question answering. Specifically, convolutional neural tensor networks [22] demonstrated the effectiveness of neural tensor composition to model relationship between question and answer pairs. Additionally, the usage of tensor

layers is also further empirically motivated in other related works in the rich history of question answering [22], sentiment analysis [26] and knowledge base completion [25]. Our work, however, is the first incorporation of neural tensor composition to the memory network architectures.

On the other hand, our second proposed model, Holo DyMemNN, is largely inspired by holographic reduced representations (HRR) [5, 19] which have seen modest exposure in recent deep learning literature. More specifically, HRR has been exploited in models such as associative LSTMs [4], holographic embeddings of knowledge graphs (HolE) [15] and Holographic Dual LSTM (HD-LSTM) [31] for QA. In this architecture, we exploit associative memory operators, namely circular correlation and circular convolution, to model the interactions between aspect and document. Notably, the properties of these operators can be also interpreted as compressed tensor products, i.e., these operators capture dyadic interactions between two vectors through summation patterns and unlike tensor layers, are essentially parameterless and therefore memory efficient. Inspired by the attractive properties of these holographic models, Nickel et al. proposed holographic embeddings of knowledge graphs [15] and adopted associative memory operators to learn embeddings of knowledge graph entities. Our work, to the best of our knowledge, is the first adaptation of holographic composition to memory networks.

3 PROBLEM FORMULATION

In this section, we formally introduce the problem of aspect-based sentiment analysis (ABSA). In this task, we explicitly distinguish between two variations of this problem. The first, aspect **term** classification considers the case where the target aspect resides in the document. For example, consider the document **D1**: 'This ice cream is really delicious'. In this case, the aspect term might be 'ice cream'. The second variation is known as aspect **category** classification in which the aspect may or may not be included in the document. For example, computing the sentiment with respect to the aspect 'food' for document **D1** would be an aspect category classification task. Intuitively, categories are often predefined aspects. For example, aspect categories like *food*, *service* and *ambiance* are common categories in the domain of restaurant reviews. Additionally, we can consider related tasks such as attitude identification and stance detection to be synonymous to aspect category classification. An example would be to identify the author's stance towards certain controversial issues (aspect categories such as feminism, abortion, etc.) in debates. Overall, ABSA can be treated as a multi-class classification problem, i.e., given a document d_i and aspect a_i, the classifier should produce a vector $f(d_i, a_i) \in \mathbb{R}^k$ where k is the number of labels. In most cases, we consider positive, negative and neutral labels. The final assigned label can be computed via $argmax\ f(d_i, a_i)$.

4 OUR DEEP LEARNING ARCHITECTURE

In this section, we introduce *Dyadic Memory Networks* (DyMemNN), our novel deep learning architecture that incorporates rich dyadic interactions between aspect and document into the memory selection operation. The overall model architecture is illustrated in Figure 1.

4.1 Input Memory Representation

Our DyMemNN architecture accepts a document d and aspect a as an input. A document is a series of words $w_i \in d$ and are assigned with an unique index. For aspect term classification, the aspect input a is similarly a sequence of one or more words. For aspect category classification, each aspect is given a separate unique index in the range of $[0, |\psi|]$ where ψ is the set of all predefined aspect categories. For the input document d, we first convert each $w_i \in d$ into *memory vectors* $m_i \in \mathbb{R}^d$ to be stored in memory. This conversion is done via an embedding look-up layer, i.e., each unique index is mapped onto the matrix $\mathbf{A} \in \mathbb{R}^{|V| \times d}$ where V is the set of all words. Similarly, this is done for the aspect a using another embedding matrix $\mathbf{B} \in \mathbb{R}^{|\psi| \times d}$ where ψ is the set of all aspect categories. Note that for aspect **term** classification, the embedding matrix $\mathbf{B}$ is of the same dimension as $\mathbf{A}$, i.e., $\mathbf{B} \in \mathbb{R}^{|V| \times d}$. In this case, we consider a neural bag-of-words adaptation of the query in which all the words in the aspect terms are summed to form the query representation.

4.2 Memory Selection Operation

Given a sequence of memory slices $\{m_i\}$ and a single aspect vector a, the objective of the memory selection operation is to select the pieces of memory slices $\{m_i\}$ that are relevant to the aspect vector $\vec{a}$. As such, the output of the memory selection operation is a vector $p \in \mathbb{L}$ which is often the output of a softmax across the input sequence. This probability vector p is then used to produce a weighted representation of the input sequence. In the standard end-to-end memory networks, the inner product $a^T m_i$ is computed for each memory slice followed by a softmax:

$$p_i = Softmax(a^T m_i) \tag{1}$$

where $Softmax(z_i) = \frac{e^{z_i}}{\sum_j e^{x_j}}$. Intuitively, inner products model the element-wise interaction between the aspect vector a and each memory slice. Tang et al. [30] extended this and adopted feed forward neural networks to model this information. This can be defined as follows:

$$p_i = Softmax(\mathbf{W}_{ff}[m_i; a] + b_{ff}) \tag{2}$$

where $\mathbf{W}_{ff}$ and b_{ff} are the parameters of the feed forward neural network and $[;]$ denotes a concatenation of aspect and memory slice.

4.3 Tensor Module

In the first variation of DyMemNN, we adopt neural tensor composition to drive the memory selection process. The tensor composition can be defined as follows:

$$z(m_i, a) = u^T tanh(m_i^T \mathbf{D}^{[1:r]}\ a + \mathbf{V}[m_i, a] + b_d) \tag{3}$$

where $\mathbf{D}^{[1:r]} \in \mathbb{R}^{n \times n \times r}$ is a tensor (3d matrix) and $tanh$ is the hyperbolic tangent function. For each slice of the tensor $\mathbf{D}$, each bilinear tensor product $m_i^T D_r\ a$ returns a scalar value to form a r dimensional vector. The other parameters, $\mathbf{V} \in \mathbb{R}^{2d \times r}$ and $u \in \mathbb{R}^r$, are in the standard form of a neural network. The final output $z(m_i, a)$ is a single scalar value. Intuitively, the tensor $\mathbf{D}$ models rich dyadic interactions between two vectors m_i and a. Additionally, this models multiple views of dyadic interactions with multiple slices and subsequently concatenates them. As such, the tensor

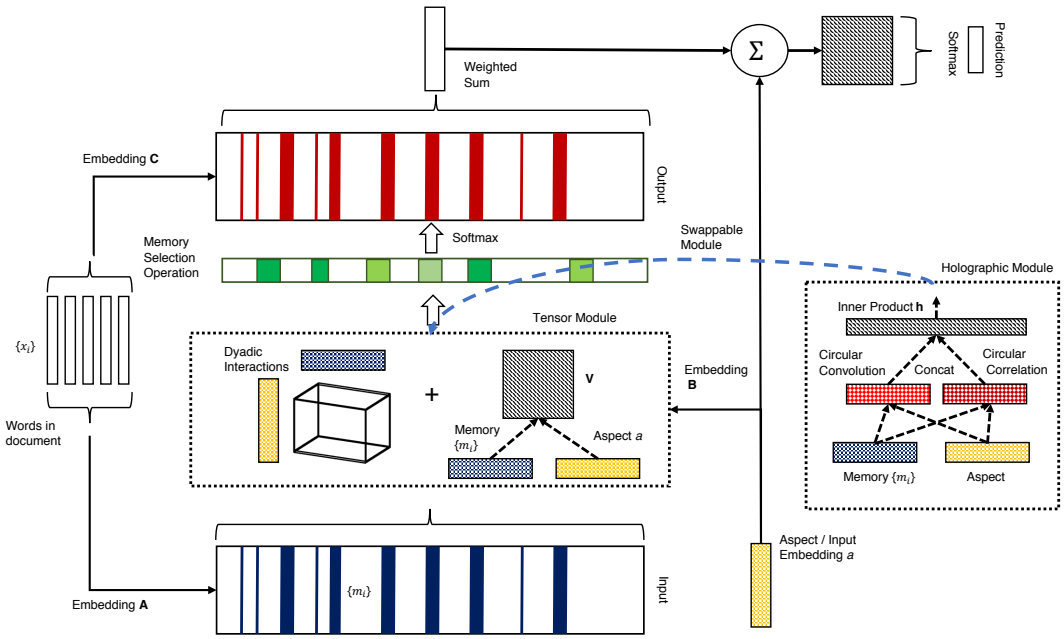

Figure 1: Diagram of our proposed DyMemNN neural architecture. This diagram illustrates the single-layered version with the tensor module. The holographic module is also depicted and can be swapped in place of the tensor module.

layer enables rich and expressive representation capability to the memory network. Finally, the memory selection operation of our Tensor DyMemNN can be rewritten as follows:

$$p_i = Softmax(u^T tanh(m_i^T \mathbf{D}^{[1:r]} a + \mathbf{V}[m_i, a] + b_d)) \quad (4)$$

4.4 Holographic Module

The second variation of DyMemNN adopts associative memory operators to drive the memory selection operation. Associative memory operators such as circular correlation and circular convolution can also be interpreted as compressed tensor products. In Holo DyMemNN, we adopt **both** circular correlation and circular convolution of vectors to learn relationships between aspect vector a and memory slices $\{m_i\}$. Each associative memory operator returns a d dimensional vector given the inputs $a \in \mathbb{R}^d$ and $m_i \in \mathbb{R}^d$.

First, we formally define our associative memory operators. Let $\circ$ denote an associative composition between two vectors. As such, circular correlation can be defined as follows:

$$m_i \circ a = m_i \star a \quad (5)$$

$$[m_i \star a]_k = \sum_{j=0}^{d-1} (m_i)_j \, a_{(k+j) \mod d} \quad (6)$$

where $\star : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ denotes the circular correlation operator, m_i is a single memory slice and a is the aspect vector. Note that we use zero indexed vectors for notational convenience. Circular correlation can be computed as follows:

$$m_i \star a = \mathcal{F}^{-1}(\overline{\mathcal{F}(m_i)} \odot \mathcal{F}(a)) \quad (7)$$

where $\mathcal{F}(.)$ and $\mathcal{F}^{-1}(.)$ are the *fast Fourier transform* (FFT) and *inverse fast Fourier transform* respectively. $\overline{\mathcal{F}(m_i)}$ denotes the complex conjugate of $\mathcal{F}(m_i)$. $\odot$ is element-wise (or Hadamard) product. The second associative memory operator, circular convolution: $* : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ can be defined as follows:

$$m_i \circ a = m_i * a \quad (8)$$

$$[m_i * a]_k = \sum_{j=0}^{d-1} (m_i)_j \, a_{(k-j) \mod d} \quad (9)$$

Circular convolution, similar to circular correlation, can be computed efficiently by:

$$m_i \star a = \mathcal{F}^{-1}(\mathcal{F}(m_i) \odot \mathcal{F}(a)) \quad (10)$$

Note that Equation (10) is the same as Equation (7) without the conjugate operator. Finally, the memory selection operation in Holo DyMemNN can be described as follows:

$$p_i = Softmax(h^T([(m_i * a); (m_i \star a)])) \quad (11)$$

where $*$ denotes circular convolution and $\star$ denotes circular correlation. $h \in \mathbb{R}^{2d}$ is a vector that maps the concatenated $[(m_i * a); (m_i \star a)]$ into a scalar value. In the following subsections, we draw some connections of Holo DyMemNN to tensor products and holographic models of associative memory.

4.4.1 Compressed Tensor Products. Both circular correlation and circular convolution can be viewed as a compressed tensor product [5, 15, 19]. In the tensor product $[m \otimes a]_{ij} = m_i a_j$, a separate element is used to store each pairwise multiplication or interaction

between m and a. In circular correlation, each element of the composed vector is a sum of multiplicative interactions over a fixed summation pattern between m_i and a. Figure 2 depicts the case where two vectors, x and y, of only three elements are composed via circular correlation and circular convolution to form the vector z. The similarity matrix denotes the standard outer (tensor) product while the same color blocks denote elements that are being 'compressed' by summation. For example in circular correlation, the value at z_1 is the summation $z_1 = x_0y_2 + x_1y_0 + x_2y_1$ which corresponds to the red squares in Figure 2 (b).

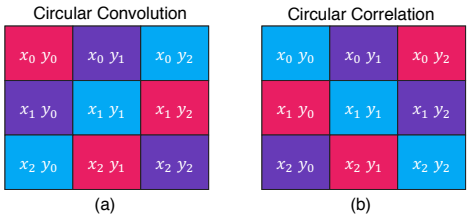

Figure 2: Circular convolution and circular correlation as compressed tensor products. Diagram is best viewed in color. Blocks of the same color denote summation (compression) patterns.

One key advantage of this composition method is that the composed vector remains at the same length of its constituent vectors. As compared to Tensor DyMemNN, Holo DyMemNN does not suffer from the quadratic increase (in d) in parameter cost. We are now also able to adapt if the dimensionality of aspect and memory slices are different. In this case, we can simply zero-pad the vectors to make them the same length. As circular correlation and circular convolution use summation patterns to compress tensor products, we are still able to compose them without much implications. However, within the scope of this paper, we consider that m and a have the same dimension.

4.4.2 Connections to Holographic Memories. In this section, we aim to draw the connection to holographic models of associative memories [5]. In holography, storage and retrieval are often emulated via a series of encoding and decoding operations. In these models, circular correlation or circular convolution can be used as the encoding operation. If circular convolution is used as the encoding operation, circular correlation becomes the decoding operation. First, a memory trace t is constructed by applying the encoding operation on two vectors, i.e., m_i and a in our case. This process is described as follows:

$$t = m_i * a$$

and subsequently a decoding operation is used to retrieve a via:

$$a' \approx m_i \star t = a * (m_i \star m_i)$$

where $m_i \star m_i \approx \delta$ is the identity element of convolution. Given the noisy vector, a', we are able to perform clean up which returns the most similar item in the stored memory t:

$$a = \arg\max_{a_i} \ a_i^{\mathsf{T}}(m_i * t)$$

Additionally, in the context of end-to-end neural network training, note that the forward operation acts as the encoding operation while the backward operation (gradient update) acts like the decoding operation. For example, consider the case where:

$$p_i = Softmax(h^T([(m_i * a)]))$$

Then, the gradients at a are as follows:

$$\frac{\partial E}{\partial a_i} = \sum_k \frac{\partial E}{\partial h_j} (m_i)_{(k-j \bmod d)}$$

where a is the aspect vector, h is the parameter of this layer and m_i is the memory slice. Note that $\sum_k \frac{\partial E}{\partial h_j} (m_i)_{(k-j \bmod d)}$ is essentially $h \star m_i$ and in this case, the parameter h acts like the memory trace t. Clearly, we see that when circular convolution is used in the neural network architecture, the gradients are being updated by circular correlation. As such, this emulates the storage and retrieval processes in holographic memory models. Intuitively, this can be seen as the parameter learning that best explains the correlation between the aspect vector and memory slice m_i.

In our Holo DyMemNN, instead of adopting a single associative memory operator as an encoder, we opt to adopt **both** circular correlation and circular convolution. There are several motivations for this choice. Firstly, this eliminates our concern on which operator to choose as an encoder. Secondly, each operator captures a slightly different view. For example, we can observe that circular correlation is asymmetric (i.e., $m_i \star a \neq a \star m_i$) while circular convolution is symmetric simply just based on the presence of the conjugate inverse operation of the former. Intuitively, combining representations from different views, i.e., exploiting both symmetric and asymmetric dyadic interactions, produces richer representations. Subsequently, we will show via an ablation study in Section 5.6.2 that this leads to better performance.

4.5 Output Memory Representation

Each input word w_i has an output vector representation c_i which is indexed from another embedding matrix $\mathbf{C} \in \mathbb{R}^{|V| \times d}$. The final output memory representation is the weighting of this probability vector c_i. As such, the output vector o is defined as follows:

$$o = \sum_i p_i c_i \tag{12}$$

where $c_i \in \mathbb{R}^d$. Intuitively, this is weighting the output vector of each word by the probability vector p and summing them up. Note that apart from the dyadic interaction (tensor or holographic) in our memory selection operation, the remainder of the network remains identical to the standard end-to-end memory network [27].

4.6 Softmax Layer

The sum of the final output vector o and the input embedding is then fed into a Softmax layer.

$$s = Softmax(\mathbf{W}_s(o + a)) \tag{13}$$

where a is the aspect embedding and o is the output vector. $\mathbf{W}_s$ is the parameter of the softmax layer. $a \in \mathbb{R}^k$ is the final output of the network where k is the number of labels.

4.7 Optimization and Learning

For optimization, our network adopts the cross entropy loss function.

$$L = -\sum_{i=1}^{N} [y_i \log s_i + (1 - y_i) \log(1 - s_i)] + \lambda \|\theta\|_2^2 \quad (14)$$

where s is the output of the softmax layer. θ contains all the parameters of the network and $\lambda \|\theta\|_2^2$ is the L2 regularization. The parameters of the network can be updated by backpropagation.

4.8 Multi-layered DyMemNN

Similar to the standard memory networks, our networks can be extended to multiple layers (or hops). Except for the first layer, the input (or aspect) embedding of the layer $k+1$ is the sum of the output o^k and input a^k from layer k. This is formally described as follows:

$$a^{k+1} = a^k + o^k \quad (15)$$

Moreover, for the sake of being prudential pertaining to the parameter size of our architecture, we adopt shared embeddings across all layers, i.e., matrices $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ are all shared across layers. For Tensor DyMemNN, tensor parameters are also shared across layers. Similarly, h is shared across layers for Holo DyMemNN as well. Following the standard memory networks, we also experimented with a linear transformation of the input representation, i.e., $a^{k+1} = \mathbf{H}(a^k + o^k)$ where $\mathbf{H} \in \mathbb{R}^{d \times d}$. The presence of this linear transformation can be considered as a hyperparameter to be tuned. Figure 3 illustrates a 2-layered DyMemNN architecture. Note that this is generalizable to k layers.

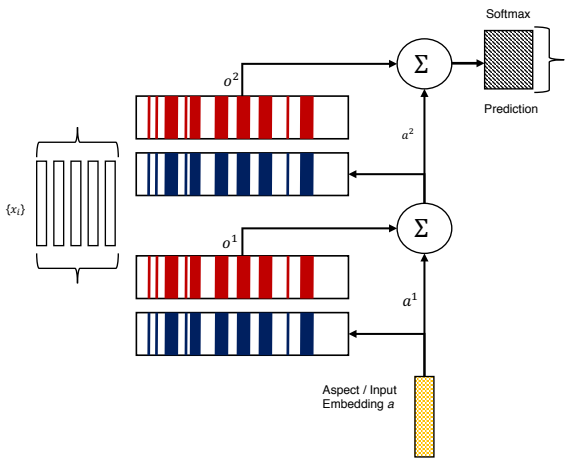

Figure 3: Illustration of 2-layered DyMemNN architecture.

5 EMPIRICAL EVALUATION

In this section, we evaluate our proposed models for aspect-based sentiment analysis. Our empirical evaluation is conducted based on **two** settings, namely aspect **term** classification and aspect **category** classification.

5.1 Datasets

In order to determine the effectiveness of our proposed models, we select a total of **six** popular benchmark datasets. These datasets can be considered as subsets of data obtained from various years and subtasks of SemEval (2014-2016). In the spirit of experimental rigor, we also included more datasets from [11]. Each dataset is used for each subtask, i.e., aspect term classification or aspect category classification. In the case where the same dataset is used for both tasks, a suffix (terms) is appended for aspect term classification and (category) is appended for aspect category classification.

5.1.1 *Datasets for Aspect Term Classification.*

- **Laptops** (term) is a dataset obtained from SemEval 2014. This dataset contains customer reviews pertaining to the computers and laptops domain.
- **Restaurants** (term) is a dataset also obtained from SemEval 2014. This dataset contains customer reviews from the restaurants domain.

5.1.2 *Datasets for Aspect Category Classification.*

- **Restaurants** (category) is a dataset obtained from SemEval 2014. Since both aspect terms and categories were provided in the restaurants dataset from SemEval 2014, we consider the same dataset but use the fixed categories such as 'food' and 'service' instead.
- **Reviews** is a merged dataset that contains reviews from both the restaurants and laptops domains obtained from **both** SemEval 2014 [21] and SemEval 2015 [20]. Only aspects that are included in both years are included.
- **Tweets** is a dataset obtained from a SemEval 2016 task pertaining to stance detection on tweets [14]. The domain of this dataset is mostly related to idealogy such as *atheism* and *feminist movement*.
- **Debates** is a dataset constructed from the Internet Argument Corpus version 2[1] which includes debates from three web forums[2]. Participants may start a debate and take a position (favor vs. against). Other users may then post arguments supporting their stance. Topics may include gun control, abortion, or death penalty. Text pre-processing includes stopword removal and tokenization by the CMU Twitter NLP tool.

The three datasets, **Debates**, **Reviews** and **Tweets** are obtained from [11]. We use the exact training, development and testing split. However, since our problem formulation does not include aspect target detection, we ignore the absent labels from the dataset from Li et al. [11]. As for the datasets **Laptops**, **Restaurants** (term) and **Restaurants** (category), we use the official testing set from SemEval 2014. Additionally, we randomly sample 500 samples from the training set as the development set. Finally, the statistics of all datasets is detailed in Table 1.

5.2 Evaluation Protocol

Whenever possible, we evaluate on both binary classification and three-way classification, (i.e., positive / negative only; and positive,

[1] https://nlds.soe.ucsc.edu/iac2.
[2] 4forums(http://www.4forums.com/political/), ConvinceMe(http://www.convinceme.net/)

Dataset	Set	#docs	#pos	#neg	#neu
Laptops (term)	Train	1813	767	673	373
	Dev	496	220	192	84
	Test	297	341‘	169	128
Restaurants (term)	Train	3102	1886	685	531
	Dev	500	278	120	102
	Test	1120	728	196	196
Restaurants (category)	Train	3018	1873	712	433
	Dev	500	306	127	67
	Test	973	657	222	94
Reviews	Train	3587	2310	1069	208
	Dev	427	274	134	19
	Test	1011	496	455	60
Debates	Train	24564	14072	10492	0
	Dev	3111	1740	1371	0
	Test	3471	1740	1731	0
Tweets	Train	2771	1349	1422	0
	Dev	212	39	173	0
	Test	1430	715	715	0

Table 1: Statistics of the six benchmark datasets.

negative and neutral) which result in a total of 10 experimental settings as the datasets Tweets and Debates only support binary classification. Since the task of sentiment analysis is commonly plagued with highly skewed class distributions, evaluation metrics such as F-score are more favorable over the accuracy metric [11].precision As such, we adopt the standard macro-averaged F-score and recall metrics in our experiments. For each class, precision is defined as $P = \frac{TP}{TP+FN}$, recall is defined as $R = \frac{TP}{TP+FP}$ and the F-score is computed by $\frac{2PR}{P+R}$. TP, TN, FN and FP are the number of true positives, true negatives, false negatives and false positives respectively. The macro-averaged F-score is the average F-score across all classes. While we report precision, recall and F-score in our experiments, we would like to emphasize that the F-score is the metric that we are looking at for the overall performance.

5.3 Compared Baselines

In our experiments, we compare our proposed models with the following state-of-the-art deep learning architectures:

- **NBOW** (Neural Bag-of-Words) is a simple summation of neural word embeddings. This summed vector is then passed into the softmax layer for prediction. This method does not consider any aspect information.
- **LSTM** (Long Short-Term Memory) is a popular neural encoder for many NLP tasks. Memory-enabled representations of the document are learned and then passed through a softmax layer for classification.
- **TD-LSTM** (Target-Dependent Long Short-Term Memory) [29] is a bidirectional LSTM that converges towards the aspect term. Subsequently the two learned representations are concatenated and passed through a softmax layer for classification.
- **AT-LSM** (Attention LSTM) is an attention-based LSTM model proposed by [33]. This model incorporates the aspect vector into the attention mechanism. As such, AT-LSTM learns to attend to different parts of the sentence based on the aspect.
- **ATAE-LSTM** (Attention LSTM with aspect embedding) is also an attention-based model that extends the AT-LSTM and was proposed by the same authors in [33]. ATAE-LSTM extends AT-LSTM by concatenating the aspect at the LSTM layer.
- **Memory Network** (MemNN) is the present state-of-the-art for the ABSA task. This model serves as the primary competitor in our ablation study. In our implementation, we adopt the feed forward neural network in the memory selection operation following [30] and since the aspect category may or may not be found within the document, we omit the location attention mechanism.

We compare the above baselines with the two variations of our proposed neural architecture, DyMemNN, namely Tensor DyMemNN and Holo DyMemNN. Note that the traditional *SVM + feature* method is omitted since Li et al. [11] has established that, even with extensive laborious feature engineering, the SVM + feature approach is outclassed by neural architectures such as LSTM or memNN.

5.4 Implementation Details

We implemented all models ourselves in Tensorflow [1] and conducted all experiments on a single NVIDIA GTX1070 GPU. For all deep learning models, we pre-initialize the word embedding layer with pretrained Glove embeddings from [16]. Specifically, we use the embeddings trained on 840 billion tokens with $d = 300$. We train all models for 25 epochs with early stopping if the performance on the development set does not improve after 5 epochs. The result reported is the model that has the highest F1-score on the development set. The batch size is fixed to 25 for all datasets except the Debates dataset in which we raised the batch size to 200 to speed up the computation time in lieu of the much larger dataset. The dimensionality of all LSTM based models is set to 300. The forget bias is set to 1 and the weight matrices are orthogonally initialized. For all models, the learning rate is tuned amongst $\{10^{-2}, 10^{-3}, 10^{-4}\}$ using the Adam optimizer [8]. L2 regularization is tuned amongst $\{10^{-4}, 10^{-5}, 10^{-6}\}$. The word embeddings are generally set to be trainable[3]. For all datasets, we capped the max sequence length to be at the *95th* percentile of all sequence lengths in the dataset. However, since the maximum sequence length of the Debates dataset is relatively much larger, we capped the max sequence length to 50 words which we found to have better performance and reduced computation time. All random initialization, if not mentioned and including oov (out-of-vocabularly) tokens in the Glove embeddings, are initialized to $U(-0.01, 0.01)$. The number of layers of memory network based models (memNN and our DyMemNN models) are tuned amongst $[1, 4]$. For Tensor DyMemNN, we initialize the parameters of the tensor layer uniformly $U(-\tau, \tau)$ where τ is tuned amongst $\tau \in \{0.01, 0.1, 0.2\}$. The number of tensor slices r is tuned amongst $\{2, 4, 6, 8, 10\}$.

5.5 Experimental Results and Discussion

In this section, we discuss the experimental results on both aspect term classification and aspect category classification. Table 2 reports the results on both binary and 3-way classification for aspect term classification. Table 3 and Table 4 report the results for aspect

[3]We also experimented with non-trainable embeddings. We found that trainable embeddings work slightly better most of the time. However, we took the best result for all models amongst the two settings.

	3-way Classification						Binary Classification					
	Laptops (term)			Restaurants (term)			Laptops (term)			Restaurants (term)		
Model	F1	P	R	F1	P	R	F1	P	R	F1	P	R
NBOW	44.07	52.88	47.32	51.18	54.18	50.83	66.00	66.38	65.71	70.26	81.23	66.93
LSTM	45.90	52.07	50.49	56.97	58.50	**58.05**	67.68	67.01	69.03	67.43	74.47	64.99
TD-LSTM	52.05	55.80	54.85	48.76	**68.26**	46.97	70.76	72.24	69.77	67.59	79.58	64.62
AT-LSTM	53.22	56.13	53.05	52.97	63.35	50.24	71.99	71.12	73.78	69.56	76.98	66.80
ATAE-LSTM	47.54	50.67	47.76	54.08	55.56	54.84	70.93	70.78	71.09	70.75	76.50	68.17
MemNN	52.15	53.68	52.86	57.49	58.85	56.55	72.50	73.39	71.82	76.46	80.74	72.92
Tensor DyMemNN	55.24†	58.08†	56.60†	58.61	60.71†	57.24	72.05	71.24	**75.42**†	79.45†	80.77	79.45†
Holo DyMemNN	**60.11**†	**60.15**†	**60.20**†	**58.82**	62.05†	57.82	**74.03**†	**75.16**†	73.19†	**79.73**†	**81.87**	**79.73**†

Table 2: Experimental results for aspect term classification on Laptops (term) and Restaurants (term). Best performance is in boldface and second best is underlined. † denotes models that significantly outperform the baseline MemNN with a t-test [35] at statistical significance level of 0.05.

	3-way Classification						Binary Classification					
	Restaurants (category)			Reviews			Restaurants (category)			Reviews		
Model	F1	P	R	F1	P	R	F1	P	R	F1	P	R
NBOW	50.76	57.77	48.49	50.21	50.30	50.18	57.08	62.08	57.84	71.94	72.50	71.92
LSTM	53.10	58.19	52.46	50.93	52.29	51.11	71.71	77.57	69.34	75.90	76.40	78.85
AT-LSTM	61.92	62.82	62.87	50.92	49.80	52.38	77.17	76.48	77.99	78.32	78.95	78.25
ATAE-LSTM	55.61	56.55	56.34	50.47	49.02	52.05	73.39	77.84	71.20	77..42	78.75	77.39
MemNN*	65.44	**69.79**	62.91	53.03	54.77	53.03	81.43	80.61	**82.43**	80.93	80.94	80.92
Tensor DyMemNN	**66.64**†	68.73	**65.17**†	**58.59**†	**59.06**†	**59.81**†	81.58	81.92	81.25	**81.66**	**81.68**	**81.65**
Holo DyMemNN	63.99	66.29	62.43	55.00	55.34	56.56†	**81.68**	**82.67**†	79.98	81.04	81.04	81.06

Table 3: Experimental results for aspect category classification on Restaurants (category) and Reviews. Best performance is in boldface and second best is underlined. † denotes models that significantly outperform the baseline MemNN with a t-test [35] at statistical significance level of 0.05.

	Binary Classification					
	Tweets			Debates		
Model	F1	P	R	F1	P	R
NBOW	66.99	67.10	66.90	53.61	56.26	54.93
LSTM	65.80	65.70	68.29	58.98	59.07	59.13
AT-LSTM	67.35	67.39	67.32	61.50	61.60	61.77
ATAE-LSTM	70.43	69.77	71.99	61.61	62.08	61.58
MemNN*	68.47	69.26	67.92	61.96	62.71	61.96
Tensor MemNN	**72.42**†	**72.11**†	**72.79**†	**66.17**†	**66.53**†	**66.07**†
Holo DyMemNN	71.24†	71.07†	71.38†	63.34	63.59	63.27

Table 4: Experimental results for aspect category classification on Tweets and Debates.

category classification. Finally, Table 5 reports the overall (averaged) F1-score across all datasets and settings. We begin by listing several important observations from our experimental results.

5.5.1 Comparison with MemNN. Across all 10 different settings, our proposed DyMemNN models consistently outperform the standard MemNN baseline. The best performance on each dataset and setting is always obtained from either Tensor DyMemNN or Holo DyMemNN. Moreover, Tensor DyMemNN and Holo DyMemNN, each individually outperforms MemNN in 9 out of 10 times. Notably, performance gains can go up to $2\% - 7\%$ in terms of F1-score. On average and overall, DyMemNNs (for both variations) are also about 2% better than MemNN. DyMemNN obtains the state-of-the-art performance on the **six** datasets and on the **ten** different experimental settings.

5.5.2 Comparison with other neural models. The performance of the standard MemNN outperforms many other neural architectures. The performance of NBOW and LSTM were often lackluster. It seems both intuitive and natural that, in ABSA tasks, modeling the relationship between aspect and word embeddings is more critical than modeling sequential dependencies. Moreover, we found that Attention-based LSTMs did not outperform the simple MemNN. To aggravate the situation, we additionally found that Attention models of Wang et al. [33] are difficult to train and optimize, i.e., they are extremely unstable and sensitive to hyperparameters.

5.5.3 Comparison between Tensor and Holo DyMemNNs. While the overall performance of Tensor DyMemNN is marginally better than Holo DyMemNN. We discuss the relative pros and cons of both models. Specifically and as seen in Table 5, Tensor DyMemNN incurs a higher parameter cost as compared to Holo DyMemNN which essentially maintains identical memory footprints with the standard MemNN. Theoretically, the cost of FFTs are log-linear, i.e., $n \log n$ [15]. The complexity of tensor computations are clearly in quadratic scale $\approx n^2$. However, FFTs and operations in complex space run much slower on GPUs as compared to highly optimized vector-matrix operations on TensorFlow. There is also overhead cost in converting vectors in and out of the complex space. As such, Tensor DyMemNN is actually significantly faster than Holo DyMemNN in our implementation which runs on GPUs. However, it is good to note that given the same unoptimized environment (i.e., only theoretically speaking), Holo DyMemNN would be significantly faster. However, at this point, we believe that Tensor DyMemNN would be a much better choice for practical applications.

5.5.4 Comparison based on the runtime and memory of all models. As a rough estimate, the time per epoch for MemNN (l=4) is $\approx 6s$ on the **Reviews** dataset. Tensor DyMemNN (l=4, r=4) runs at $\approx 9s$ while Holo DyMemNN (l=4) runs at $\approx 55s$. The time cost for LSTM is $\approx 9s$ while AT-LSTM and ATAE-LSTM are $\approx 11/12s$. The computation time of Tensor DyMemNN is in fact still similar and competitive to the baseline LSTM. Pertaining to memory footprint of the compared neural models, MemNN is highly compact and

memory efficient with about $\approx 3-4$ times less parameters as compared to the LSTM based models. Tensor DyMemNN incurs a cost of quadratic scale with respect to n^2 which is equivalent to about 90K parameters given a dimensionality of 300. Given a modest r value such as $r = 4$ (number of tensor slices), the Tensor DyMemNN would have approximately similar number of parameters to LSTM. Given the significantly improved performance, Tensor DyMemNN, safely speaking, provides a better performance gain per parameter as compared to LSTM.

Model	F1	# Params	Aspect
NBOW	58.2	2K	✗
LSTM	61.4	724K	✗
TD-LSTM	63.6	724K	✓
AT-LSTM	62.4	1.1M	✓
ATAE-LSTM	61.7	1.4M	✓
MemNN	66.9	101K	✓
Tensor DyMemNN	**69.2**†	$\approx$ (101K + $r\times$ 90K)*	✓
Holo DyMemNN	69.0†	101K	✓

Table 5: Overall performance (F1) of all compared models on all the six benchmark datasets with both 2 and 3 way classification. Aspect column denotes if the model has benefited from aspect information. $*$ denotes the variable number of parameters across datasets. About $\approx 90K$ parameters are added for each extra tensor slice.

5.6 Effect of Hyperparameters

In this section, we aim to study two important research questions pertaining to our models. Specifically, they are **RQ1:**'What are the effects of the number of tensor slices on performance?' and **RQ2:**'What are the benefits of using both associative operators as compared to using only one?'.

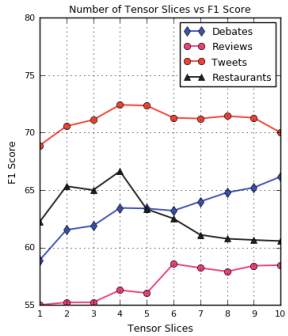

Figure 4: Effect of varying tensor slices r on aspect category classification.

5.6.1 Effect of Tensor Slices r on Performance. Figure 4 reports the test performance varied across different r values with $l = 4$. Note that the optimal tensor slice of the development set and test set were the same. This was conducted across all datasets for aspect category classification with 3-way classification whenever possible. The outcomes of this study allow us to understand several points as follows: First, increasing r helps mainly when the dataset is large (e.g., the **Debates** dataset). On the other hand, increasing r too high when the dataset is small **may** hurt performance (e.g., the **Restaurants** dataset). This is intuitive as the tensor module with a large r introduces a significant amount of parameters to the model and incurs a risk of overfitting. On this note, we can offer a general rule of thumb for training our models, i.e., set $r = 4$ or $r = 5$ unless the dataset is huge. In that case, going upwards towards even $r = 10$ might be reasonable.

5.6.2 Ablation Study on Associative Memory Operators. Table 6 reports the results of our ablation study on the influence of associative memory operators. We make several important observations. Firstly, using both circular correlation and circular convolution leads to improved performance. This justifies the design of Holo DyMemNN. Secondly, circular correlation (asymmetric) representations are more beneficial over circular convolution. We would like to advocate the usage of both operators when possible. However, if only one has to be selected (perhaps to reduce computation time), choosing circular correlation would be more sensible.

Dataset	Remove Conv $*$	Remove Corr $\star$	Full Holo
Laptops (t)	54.93 (−5.18%)	53.98 (−6.13%)	60.11
Restaurants (t)	61.68 (+2.86%)	55.94 (−2.88%)	58.82
Restaurants (c)	62.02 (−1.97%)	61.95 (−2.04%)	63.99
Reviews	54.65 (−0.35%)	53.89 (−1.11%)	55.00
Tweets	69.79 (−1.45%)	68.89 (−2.35%)	71.24
Debates	62.34 (−1.00%)	61.07 (−2.27%)	63.34

Table 6: Ablation studies on associative memory operators for Holo DyMemNN. It reports F1 scores across all datasets when only circular correlation or circular convolution is used. Full Holo describes the full holographic architecture.

Aspect Term	Document
MemNet	
place	Nice place but the burgers were awful!
burgers	Nice place but the burgers were awful!
Tensor DyMemNN	
place	Nice place but the burgers were awful!
burgers	Nice place but the burgers were awful!
Holo DyMemNN	
place	Nice place but the burgers were awful!
burgers	Nice place but the burgers were awful!

Table 7: Qualitative analysis of attention vector p. The intensity of the color (red) denotes the strength of the attention weights.

5.7 Qualitative Analysis

In this section, we inspect the vector p (also known as the attention vector) that is used to generate the weighted representation. The purpose and meaning behind this experiment, which forms our third research question, is as follows: **RQ3:** 'Is there any meaningful and observable differences between the attention vector of MemNN and DyMemNN?'. In order to do so, we extracted p, the attention vector from the models MemNN, Tensor DyMemNN ($r = 4$) and Holo DyMemNN. Table 7 depicts the outcome of a single example.

We observe that DyMemNN generally produces more *focused* attentions over MemNN. The attention is divided quite evenly

amongst words for MemNN. In the given example, both variations of DyMemNN have strong attention weights towards the appropriate words and are able to adaptively switch attentions based on the query or aspect term. On the other hand, MemNN has weak attentions spread out across the document.

6 CONCLUSION

In this paper, we proposed a new neural architecture for ABSA. We proposed two variations of DyMemNN, namely Tensor DyMemNN and Holo DyMemNN, which model dyadic interactions between aspect and document. Modeling rich interactions between aspect and document results in improved performance. We show that the two models of DyMemNN demonstrate the state-of-the-art performance on six benchmark datasets. Moreover, the two models consistently outperform the baseline MemNN on both aspect term and aspect category classification tasks.

7 ACKNOWLEDGEMENTS

The authors would like to thank anonymous reviewers for their hardwork and comments.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). Software available from tensorflow.org.

[2] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale Simple Question Answering with Memory Networks. *CoRR* abs/1506.02075 (2015). http://arxiv.org/abs/1506.02075

[3] James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2016. Quasi-Recurrent Neural Networks. *CoRR* abs/1611.01576 (2016).

[4] Ivo Danihelka, Greg Wayne, Benigno Uria, Nal Kalchbrenner, and Alex Graves. 2016. Associative Long Short-Term Memory. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. 1986–1994.

[5] D. Gabor. 1969. Associative Holographic Memories. *IBM J. Res. Dev.* 13, 2 (March 1969), 156–159. DOI: http://dx.doi.org/10.1147/rd.132.0156

[6] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[7] Nobuhiro Kaji and Masaru Kitsuregawa. 2007. Building Lexicon for Sentiment Analysis from Massive Collection of HTML Documents.. In *EMNLP-CoNLL*. 1075–1083.

[8] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[9] Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. NRC-Canada-2014: Detecting Aspects and Sentiment in Customer Reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014*. 437–442.

[10] Svetlana Kiritchenko, Xiaodan Zhu, and Saif M. Mohammad. 2014. Sentiment Analysis of Short Informal Texts. *J. Artif. Intell. Res. (JAIR)* 50 (2014), 723–762. DOI: http://dx.doi.org/10.1613/jair.4272

[11] Cheng Li, Xiaoxiao Guo, and Qiaozhu Mei. 2017. Deep Memory Networks for Attitude Identification. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*. 671–680.

[12] Pengfei Liu, Shafiq R. Joty, and Helen M. Meng. 2015. Fine-grained Opinion Mining with Recurrent Neural Networks and Word Embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*. 1433–1443. http://aclweb.org/anthology/D/D15/D15-1168.pdf

[13] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[14] Saif M Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. Semeval-2016 task 6: Detecting stance in tweets. *Proceedings of SemEval* 16 (2016).

[15] Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. 2016. Holographic Embeddings of Knowledge Graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*. 1955–1961.

[16] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. 1532–1543.

[17] Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. 2017. NeuPL: Attention-based Semantic Matching and Pair-Linking for Entity Disambiguation. In *Proceedings of the 2017 ACM CIKM International Conference on Information and Knowledge Management, Singapore November 6 - November 10, 2017*. DOI: http://dx.doi.org/10.1145/3132847.3132963

[18] Tony Plate. 1992. Holographic Recurrent Networks. In *Advances in Neural Information Processing Systems 5, [NIPS Conference, Denver, Colorado, USA, November 30 - December 3, 1992]*. 34–41.

[19] Tony A. Plate. 1995. Holographic reduced representations. *IEEE Trans. Neural Networks* 6, 3 (1995), 623–641. DOI: http://dx.doi.org/10.1109/72.377968

[20] Maria Pontiki, Dimitrios Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. SemEval-2015 Task 12: Aspect Based Sentiment Analysis. (????).

[21] Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 Task 4: Aspect Based Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014*. 27–35.

[22] Xipeng Qiu and Xuanjing Huang. 2015. Convolutional Neural Tensor Network Architecture for Community-Based Question Answering. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*. 1305–1311.

[23] Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 675–682.

[24] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664* (2015).

[25] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*. 926–934.

[26] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. Citeseer.

[27] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-To-End Memory Networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. 2440–2448.

[28] Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* (2015).

[29] Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016. Effective LSTMs for Target-Dependent Sentiment Classification. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*. 3298–3307.

[30] Duyu Tang, Bing Qin, and Ting Liu. 2016. Aspect Level Sentiment Classification with Deep Memory Network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. 214–224.

[31] Yi Tay, Minh C. Phan, Anh Tuan Luu, and Siu Cheung Hui. 2017. Learning to Rank Question Answer Pairs with Holographic Dual LSTM Architecture. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*. 695–704. DOI: http://dx.doi.org/10.1145/3077136.3080790

[32] Zhiyang Teng, Duy-Tin Vo, and Yue Zhang. 2016. Context-Sensitive Lexicon Features for Neural Sentiment Analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. 1629–1638. http://aclweb.org/anthology/D/D16/D16-1169.pdf

[33] Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for Aspect-level Sentiment Classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. 606–615.

[34] Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic Memory Networks for Visual and Textual Question Answering. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. 2397–2406. http://jmlr.org/proceedings/papers/v48/xiong16.html

[35] Yiming Yang and Xin Liu. 1999. A Re-Examination of Text Categorization Methods. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 15-19, 1999, Berkeley, CA, USA*. 42–49. DOI: http://dx.doi.org/10.1145/312624.312647

[36] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*.