A Hybrid Approach for Aspect-Based Sentiment Analysis Using a Lexicalized Domain Ontology and Attentional Neural Models

Olaf Wallaart
olafwallaart@student.eur.nl

Flavius Frasincar
frasincar@ese.eur.nl

Erasmus University Rotterdam
the Netherlands

Contents

Motivation

- Explosion of opinionated text on the Web (e.g., reviews)
- Difficult to manually determine opinions with respect to an entity of interest
- **Sentiment Analysis (SA)**: automatic computation of the sentiment (e.g., positive, neutral, or negative) expressed in a piece of text (related to the entity of interest)
- **Aspect-Based Sentiment Analysis (ABSA)**: automatic computation of the sentiment expressed for an aspect in a piece of text (related to the entity of interest)
  - Review-level
  - **Sentence-level** [our focus here]

Motivation

ABSA Tasks

1. Target Extraction: extracting the target word or set of words present in text
2. Aspect Detection: detecting the aspects from text (aspects can be broader than targets)
   - Explicit Aspect Detection: aspects have associated targets in text
   - Implicit Aspect Detection: aspects do not have associated targets in text (they are implied from text)
3. **Sentiment Classification**: computing the sentiment associated to an (implicit or explicit) aspect [our focus here]

Related Work

ABSA Solutions

- Knowledge-Based Reasoning: using lexicalized sentiment domain ontologies one can infer the sentiment attached to an aspect in context, e.g., Ont (Schouten and Frasincar, 2018)
- Machine Learning: using machine learning approaches based on feature vectors, e.g., Bag-of-Words (BoW), which includes aspect and sentence sentiment score (Schouten and Frasincar, 2018)
  - Deep Learning: learn feature representations and consider word order, e.g., CABASC (Liu et al., 2018), LCR-Rot (Zheng and Xia, 2018)
- **Hybrid**: mix knowledge-based reasoning with machine learning, e.g., BoW+Ont (Schouten and Frasincar, 2018)
  - **Two-Step Hybrid**: first knowledge-based reasoning and then machine learning as backup, e.g., Ont+BoW (Schouten and Frasincar, 2018), which uses an SVM classifier as backup

Main Ideas

- We propose a two-step hybrid approach that performs first Knowledge-Based Reasoning and then Deep Learning as backup dubbed **A Hybrid Approach for Aspect-Based Sentiment Analysis (HAABSA)**
- We reuse Knowledge-Based Reasoning from Ont+BoW, but extend LCR-Rot for Deep Learning:
  - Inverting the attention order in the rotary attention mechanism
  - Performing multiple hops in the rotary attention mechanism

Methodology

The methodology is based on two main steps:

1. **Knowledge-Based Reasoning**
2. **Deep Learning**

More precisely:

1.1 Compute the sentiment of each word in a sentence that is related to the current aspect based on a lexicalized domain sentiment ontology

1.2 If only positive sentiment found then classify the aspect as positive

1.3 If only negative sentiment found then classify the aspect as negative

2 If both positive and negative sentiment found or no sentiment found then apply the Neural Attention Model to classify the aspect as positive, negative, or neutral

Ontology

The onology has three main classes:

- *SentimentMention*: specifies the mentions of sentiment
- *SentimentValue*: specifies the polarity which can be *Positive* or *Negative*
- *AspectMention*: specifies the mentions of aspects

and two main (annotation) properties:

- *lex*: relates a mention to a lexical representation
- *aspect*: relates a sentiment mention to an aspect

In addition:

- The *Neutral* sentiment is not specified due to its ambiguous semantics
- We consider negation in two cases:
  - Using the dependency relation the current word is related to a *Negator*
  - One of the preceding three words with respect to the current word is a *Negator*

Sentiment Mention Types

There are three sentiment mention types:

- *Type 1*: generic sentiment mention, which has the same sentiment value for all aspects
  - e.g., "awesome" is always *Positive* (unless sarcasm present, which we do not consider here)
- *Type 2*: aspect-dependent sentiment, which has the same sentiment value for some aspects (extra check for matching the current aspect needed)
  - e.g., "delicious" is *Positive* for *SustenanceMention* (food and drinks) but does not apply to *ServiceMention*
- *Type 3*: context-dependent sentiment, which has different sentiment values for different aspects (extra check for matching the current aspect needed)
  - New axioms built based on a sentiment mention linked by a dependency relation to the current aspect
  - e.g., "cold" + "beer" is *Positive* but "cold" + "pizza" is *Negative*

Neural Attention Model

The Neural Attention Model dubbed **Left-Center-Right Separated Neural Network with Rotatory Attention (LCR-Rot)** has five main parts:

1. *Word Embeddings*: represents the sentence a sequence of word embeddings
2. *Context-Based Word Embeddings*: represents the sentence as a sequence of context-based word embeddings
3. *Target2Context Attention Mechanism*: computes the target-aware left/right context representation
4. *Context2Target Attention Mechanism*: computes the left/target context-aware target representation
5. *Classification Module*: classifies an aspect for sentiment (positive, neutral, or negative)

Word Embeddings

- $S = \{s_1, s_2, ..., s_N\}$ be an input sentence of length N containing words s_i
- We split the sentence into three disjoint and consecutive parts, and get:
  - *Left context*: $S^l = [s_1^l, s_2^l, \ldots, s_L^l]$, where L is the length of the left context
  - *Target phrase*: $S^t = [s_1^t, s_2^t, \ldots, s_M^t]$, where M is the length of the target phrase (the relevant aspect)
  - *Right context*: $S^r = [s_1^r, s_2^r, \ldots, s_R^r]$, where R is the length of the right context
- The corresponding word embeddings are:
  - *Left context*: $E^l = [e_1^l, e_2^l, \ldots, e_L^l]$
  - *Target phrase*: $E^t = [e_1^t, e_2^t, \ldots, e_M^t]$
  - *Right context*: $E^r = [e_1^r, e_2^r, \ldots, e_R^r]$
    where $e_n \in \mathbb{R}^d$ and d is the dimension of the word embedding (usually $d = 300$)

Context-Based Word Embeddings

We apply three bi-directional long-short-term-memory (Bi-LSTM) modules (with 300 hidden units each) on the previous word embeddings:

- Left Bi-LSTM: computes the left context hidden values $H^l = [h_1^l, h_2^l, \ldots, h_L^l]$
- Target Bi-LSTM: computes the target phrase hidden values $H^t = [h_1^t, h_2^t, \ldots, h_M^l]$ for target phrase
- Right Bi-LSTM: computes the right context hidden values $H^r = [h_1^r, h_2^r, \ldots, h_R^r]$
  where $h_n \in \mathbb{R}^{2xd}$ obtained by concatenating hidden values in each direction

Bi-LSTM Advantages

- Remember relevant information for a long period of time
- Keep contextual information in both directions

Target2Context Attention Mechanism

Computes the **target-aware left context representation** as follows:

- Determine a target representation using an average pooling layer:

$$\underset{2d\times 1}{r^{t_p}} = \text{avg\_pooling}([\underset{2d\times 1}{h_1^t}, \underset{2d\times 1}{h_2^t}, \ldots, \underset{2d\times 1}{h_M^t}]) \quad (1)$$

- Determine the left context word attention values using a bilinear form involving hidden values h_i^l, for $i = 1, \ldots, L$, and r^{t_p}:

$$\underset{1\times 1}{f(h_i^l, r^{t_p})} = \tanh(\underset{1\times 2d}{h_i^{l'}} \times \underset{2d\times 2d}{W_c^l} \times \underset{2d\times 1}{r^{t_p}} + \underset{1\times 1}{b_c^l}) \quad (2)$$

where W_c^l is a weight matrix and b_c^l is a bias

Target2Context Attention Mechanism

- Squash the attention values to values between 0 and 1 using the softmax function:

$$\alpha_i^l = \frac{\exp(f(h_i^l, r^{t_p}))}{\sum_{j=1}^{L} \exp(f(h_j^l, r^{t_p}))} \tag{3}$$

- Determine the target-aware left context representation as an attention weighted average of word hidden values:

$$\underset{2d\times 1}{r^l} = \sum_{i=1}^{L} \underset{1\times 1}{\alpha_i^l} \times \underset{2d\times 1}{h_i^l} \tag{4}$$

By following Equations (2)-(4) in a similar way we can obtain r^r for the **target-aware right context representation**

Context2Target Attention Mechanism

Computes an improved target representation as follows:

- Determine the target word attention values using a bilinear form involving hidden values h_i^t, for $i = 1, \ldots, M$, and r^l:

$$f(\underset{1\times 1}{h_i^t}, r^l) = \tanh(\underset{1\times 2d}{h_i^{t\prime}} \times \underset{2d\times 2d}{W_t^l} \times \underset{2d\times 1}{r^l} + \underset{1\times 1}{b_t^l}), \tag{5}$$

  where W_t^l is a weight matrix and b_t^l is a bias

- Squash the attention values to values between 0 and 1 using the softmax function:

$$\alpha_i^{t_l} = \frac{\exp(f(h_i^t, r^l))}{\sum_{j=1}^{M} \exp(f(h_j^t, r^l))}. \tag{6}$$

Context2Target Attention Mechanism

- Determine the **left context-aware target representation** as an attention weighted average of word hidden values:

$$\underset{2d\times 1}{r^{t_l}} = \sum_{i=1}^{M} \underset{1\times 1}{\alpha_i^{t_l}} \times \underset{2d\times 1}{h_i^t}\,, \tag{7}$$

By following Equations (5)-(7) in a similar way we can obtain the **right context-aware target representation**, r^{t_r}

Classification Module

- The **sentence representation** is obtained by concatenating the target-aware left context representation r^l, target-aware right context representation r^r, and the two sides context-aware target representations, r^{t_l} and r^{t_r}:

$$\underset{8d\times 1}{v} = [\ \underset{2d\times 1}{r^l}\ ;\ \underset{2d\times 1}{r^{t_l}}\ ;\ \underset{2d\times 1}{r^{t_r}}\ ;\ \underset{2d\times 1}{r^r}\ ] \tag{8}$$

- The sentence representation vector is converted by a linear layer to a vector of size C, where C is the number of different sentiment categories, and the obtained vector is then fed into a softmax layer to predict the sentiment polarity of the target phrase:

$$\underset{|C|\times 1}{p} = \text{softmax}(\ \underset{|C|\times 8d}{W_c}\ \times\ \underset{8d\times 1}{v}\ +\ \underset{|C|\times 1}{b_c}\ ) \tag{9}$$

where p is a conditional probability distribution, W_c is a weight matrix, and b_c is a bias

Model Training

- We minimize the cross-entropy loss function defined as:

$$\underset{1\times 1}{L} = -\sum_{j} \underset{1\times|C|}{y_j'} \times \underset{|C|\times 1}{\log(p_j)} + \lambda\|\Theta\|^2 \tag{10}$$

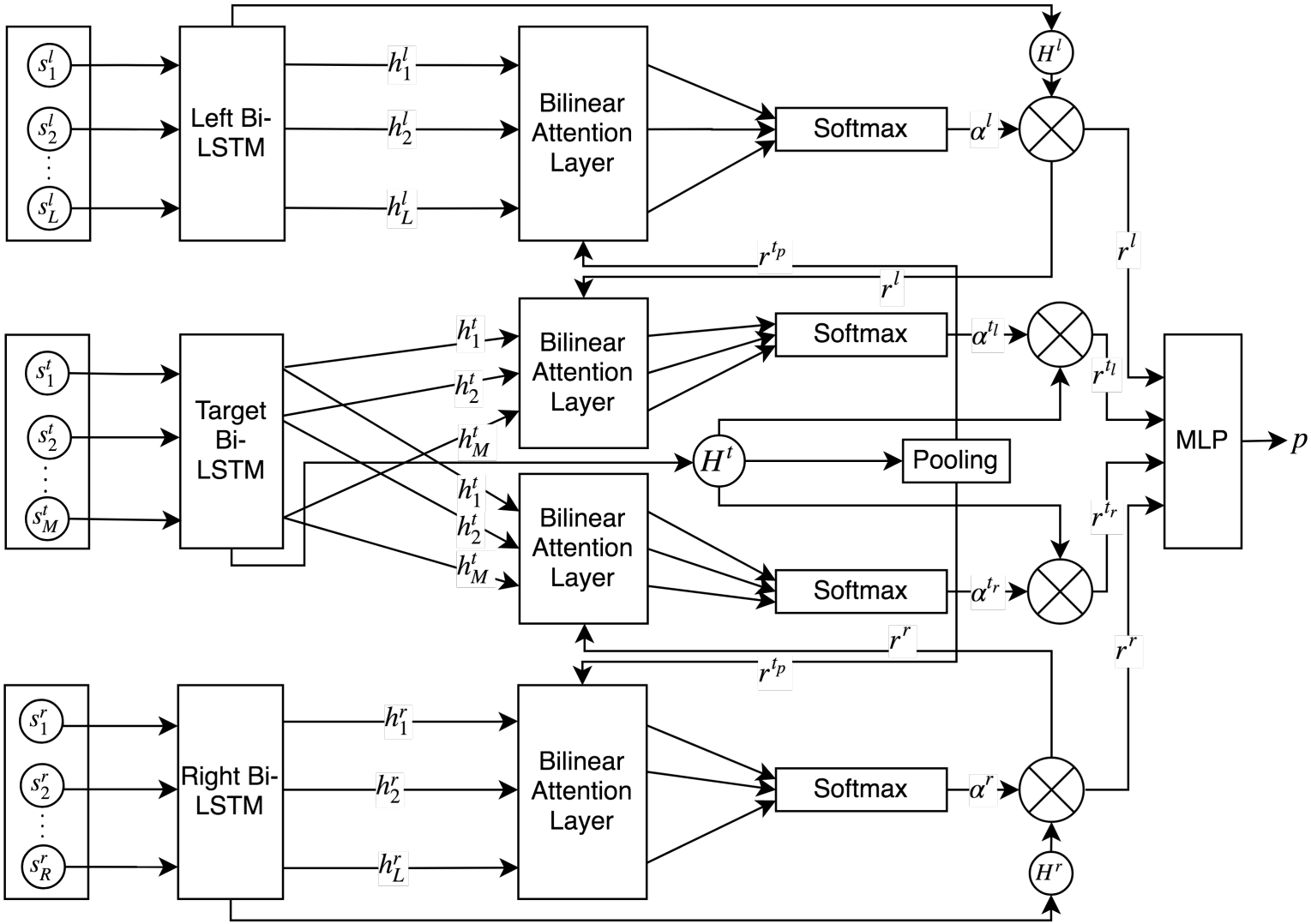
where

- y_j' is a vector that contains the true sentiment value for the j-th training opinion
- p_j is a vector containing the predicted sentiment for the j-th training opinion
- λ is the weight of the L_2-regularization term
- Θ is the parameter set which contains $\{W_c^l,\ b_c^l,\ W_c^r,\ b_c^r,\ W_t^l,\ b_t^l,\ W_t^r,\ b_t^r,\ W_c,\ b_c\}$ and the three Bi-LSTMs parameters

Model Training

- For loss minimization we use backward error propagation
- We intialize weight matrices by a uniform distribution $U(-0.1, 0.1)$ and all bias are set to zero, as is done by Zheng and Xia (2018)
- To update the weights and biases we use stochastic gradient descent with momentum
- The dropout technique is applied to all hidden layers to avoid overfitting
- The following hyperparameters are tuned on 20% of the training data (80% of the training data is used for model building) using a Tree-structured Parzen Estimator (TPE) algorithm:
  - the learning rate
  - the L_2-regularization term (λ)
  - the dropout rate
  - the momentum term

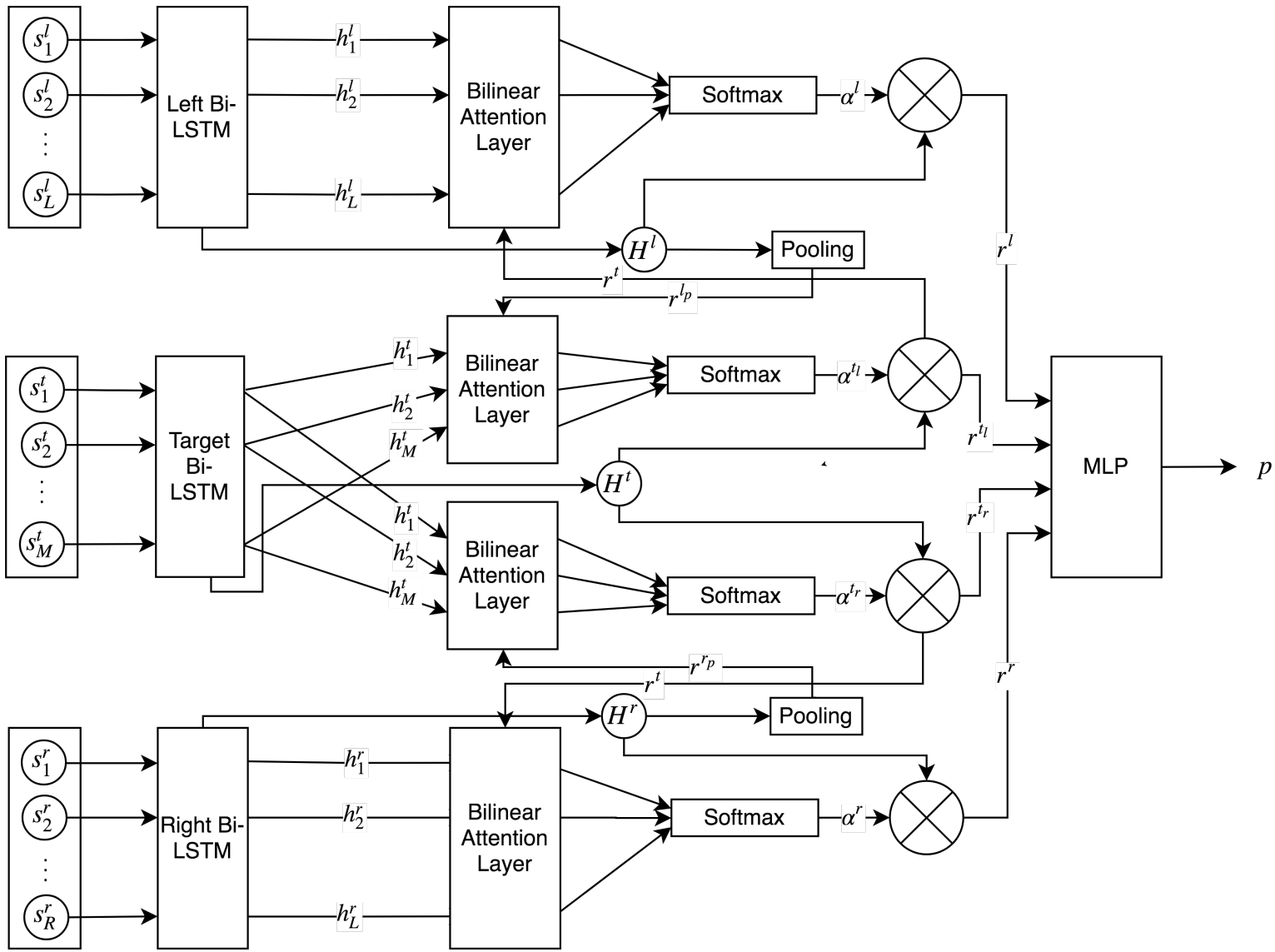
LCR-Rot Architecture



LCR-Rot-inv Architecture

- Inverting the attention order in the rotary attention mechanism:
  1. Apply context2target attention mechanism
  2. Apply target2context attention mechanism
- We start with two context pooling layers:
  - Left context pooling
  - Right context pooling

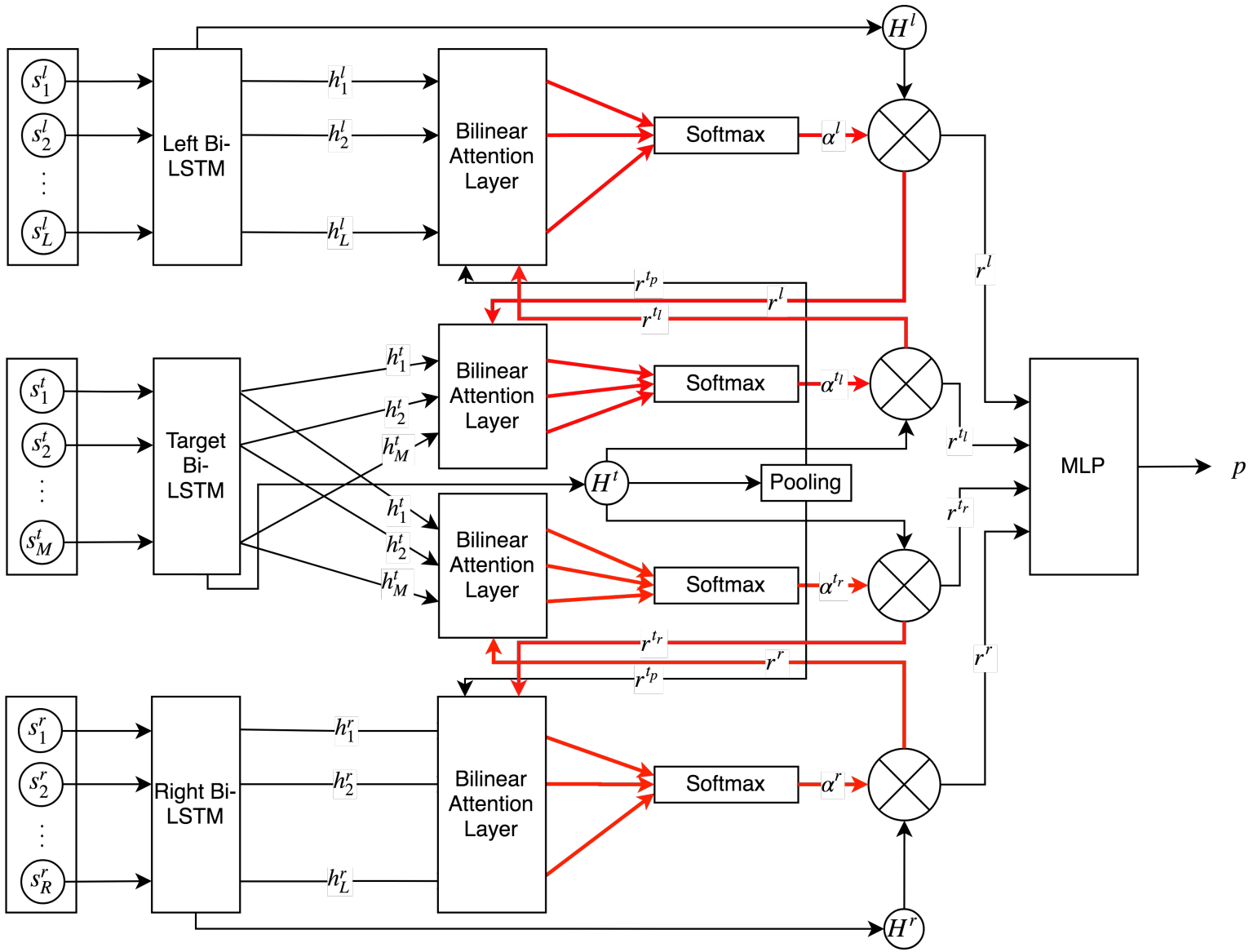
LCR-Rot-inv Architecture



LCR-Rot-hop Architecture

- Performing multiple hops in the rotary attention mechanism
- Improve the interaction between aspects and contexts
- Alternate between two steps:
  1. LCR-Rot (start with the pooling of target words)
  2. LCR-Rot-inv (no need for pooling contexts, as context representations have been previously computed)

LCR-Rot-hop Architecture



Data

- Data made available by the International Workshop on Semantic Evaluation (SemEval)
- SemEval 2015 Task 12 (Slot 3) and SemEval 2016 Task 5 (Subtask 1) each have:
  - Restaurant Domain English Training Data
  - Restaurant Domain English Gold Annotations Data
- Web restaurant reviews
- Example:

```
<sentence id="1154550:1">
 <text>The place is small and cramped but the food is
       fantastic.</text>
  <Opinions>
   <Opinion target="place" category="AMBIENCE#GENERAL"
            polarity="negative" from="4" to="9" />
   <Opinion target="food" category="FOOD#QUALITY"
            polarity="positive" from="39" to="43" />
</Opinions>
</sentence>
```

Data

- Polarity distributions in train and test data sets:

	Negative		Neutral		Positive		Total	
	Freq.	%	Freq.	%	Freq.	%	Freq.	%
SemEval-2016 train data	488	26.0	72	3.8	1319	70.2	1879	100
SemEval-2016 test data	135	20.8	32	4.9	483	74.3	650	100

- Data is skewed towards positive sentiment (around 70%)
- Category counts and frequencies in the data set:

Categories	Train data		Test data	
	Freq.	%	Freq.	%
AMBIENCE#GENERAL	226	12.03	59	9.08
DRINKS#PRICES	20	1.06	4	0.62
DRINKS#QUALITY	44	2.34	22	3.38
DRINKS#STYLE_OPTIONS	32	1.7	11	1.69
FOOD#PRICES	70	3.73	22	3.38
FOOD#QUALITY	766	40.77	283	43.54
FOOD#STYLE_OPTIONS	115	6.12	51	7.85
LOCATION#GENERAL	22	1.17	10	1.54
RESTAURANT#GENERAL	185	9.85	58	8.92
RESTAURANT#MISCELLANEOUS	49	2.61	18	2.77
RESTAURANT#PRICES	26	1.38	5	0.77
SERVICE#GENERAL	324	17.24	107	16.46

- The most dominant aspect is "FOOD#QUALITY" (around 40%)

Data

Data Cleaning:

- All words are converted to lowercase and lemmatized
- Sentences containing implicit aspects (*target*="*NULL*") are not considered (as we need the targets)
- Sentences without *Opinions* are also excluded (no use)

Word Emebeddings:

- GloVe word embedding vectors: 1.9 million vocabulary size with 300-dimensional vectors
- Words that do not appear in the GloVe vocabulary are randomly initialized by a normal distribution $N(0, 0.05^2)$ as by Zheng and Xia (2018) (only 3.6% of the words are not in the GloVe vocabulary, these words are often names of restaurants, jargon, or slang)

Evaluation

- Training is performed on the training data and testing is done on the official test data
- The evaluation metric is classification accuracy (same as used in the SemEval competition)
- Reference models:
  - *Ont*: knowledge-based reasoning with backup majority polarity (Schouten and Frasincar, 2018)
  - *BoW*: bag-of-words method combined with an SVM classifier to determine sentiment (Schouten and Frasincar, 2018)
  - *CABASC*: neural network that contains a context attention-based memory module (Liu et al., 2018)
  - *Ont+BoW*: two-step approach where an ontology method is first used and as backup the bag-of-words method is used (Schouten and Frasincar, 2018)
  - *Ont+CABASC*: two-step approach where an ontology method is first used and as backup the CABASC method is used (new baseline)

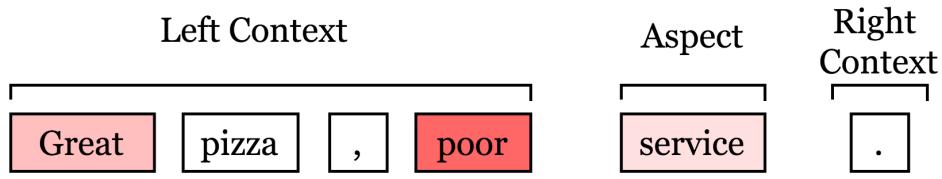
Evaluation

	SemEval-2015				SemEval-2016			
	out-of-sample	in-sample	cross-validation		out-of-sample	in-sample	cross-validation	
	acc.	acc.	acc.	st. dev.	acc.	acc.	acc.	st. dev.
Ont	65.8%	79.7%	79.7%	0.0183	78.3%	75.3%	75.3%	0.0152
BoW	76.2%	91.0%	87.9%	0.0311	83.2%	89.3%	84.5%	0.0254
CABASC	76.6%	85.8%	87.1%	0.0138	84.6%	79.2%	84.0%	0.0218
LCR-Rot	**78.4%**	86.2%	88.0%	0.0144	86.9%	92.9%	85.8%	0.0214
LCR-Rot-inv	77.1%	85.2%	88.1%	0.0146	86.5%	93.9%	85.5%	0.0161
LCR-Rot-hop	**78.4%**	88.6%	87.6%	0.0181	**87.7%**	86.3%	85.6%	0.0169
Ont+BoW	79.5%	86.9%	83.5%	0.0308	85.6%	86.7%	85.7%	0.0329
Ont+CABASC	79.6%	84.3%	83.2%	0.0138	85.9%	82.3%	85.5%	0.0298
Ont+LCR-Rot	**80.6%**	84.5%	83.7%	0.0144	87.0%	88.3%	86.3%	0.0323
Ont+LCR-Rot-inv	79.9%	89.0%	83.7%	0.0146	86.6%	88.7%	86.2%	0.0296
Ont+LCR-Rot-hop	**80.6%**	85.7%	83.5%	0.0298	**88.0%**	86.7%	86.2%	0.0308

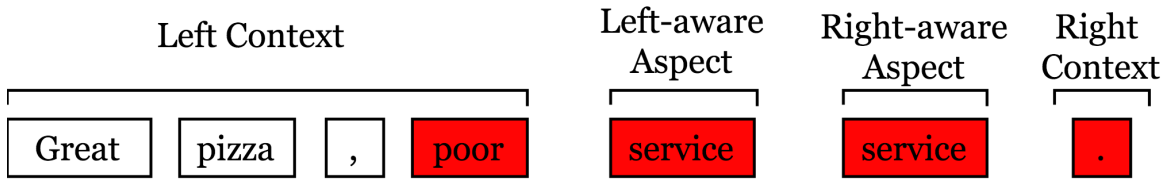
- Ont does not perform well (the ontology predicts in 60% of the cases, the majority classifier [poor predictor] predicts in 40% of the cases)
- LCR-Rot outperforms CABASC by 1.8%-2.3%
- LCR-Rot-inv is not better than LCR-Rot
- LCR-Rot-hop (3 hops) outperforms LCR-Rot by 0.0%-0.8%
- Hybrid methods are better than original methods (ontology provides additional information)
- Ont-LCR-Rot-hop outperforms LCR-Rot-hop (3 hops) by 2.2%-0.3%

Evaluation

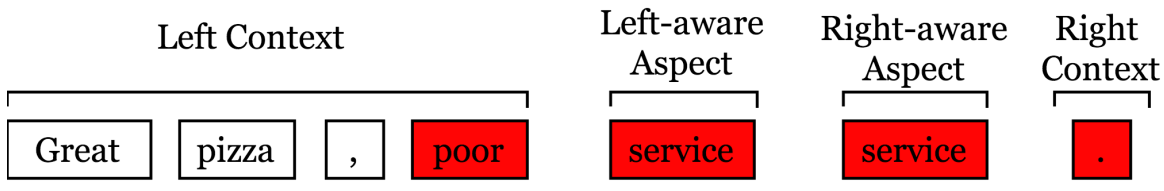
Attention visualizations of the LCR-Rot, LCR-Rot-hop, and CABASC models for the phrase 'Great pizza, poor service' for aspect 'service':



(a) CABASC attention visualization



(b) LCR-Rot attention visualization

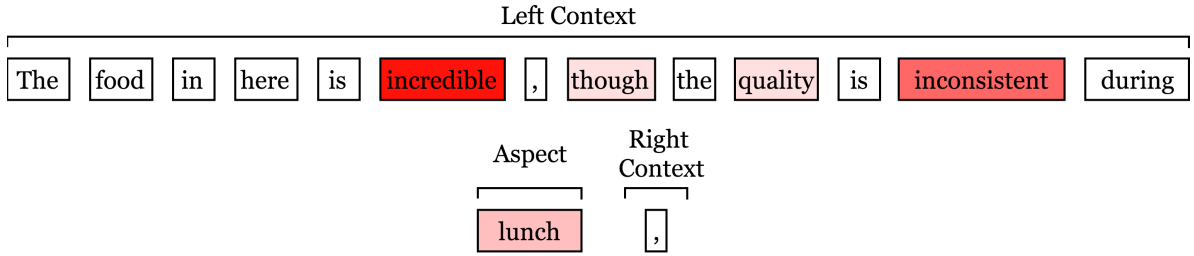


(c) LCR-Rot-hop attention visualization

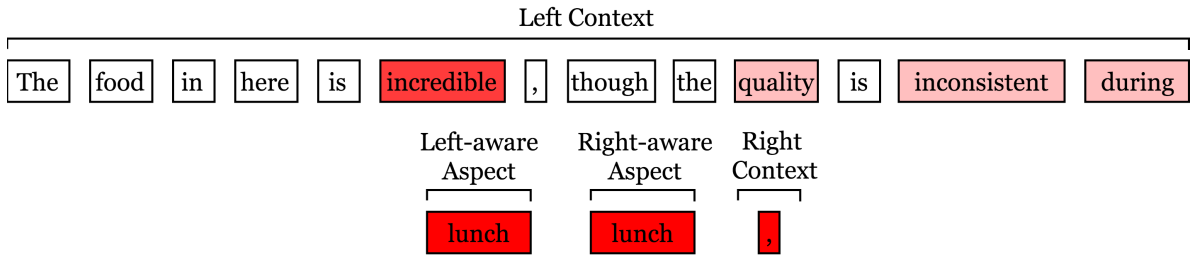
- Red color indicates word attention (dark color means high attention)
- CABASC wrongfully pays attention to the word 'great' (which belongs to aspect 'pizza') [incorrect classification]
- LCR-Rot and LCR-Rot-hop rightfully give more attention to 'poor' [correct classification]

Evaluation

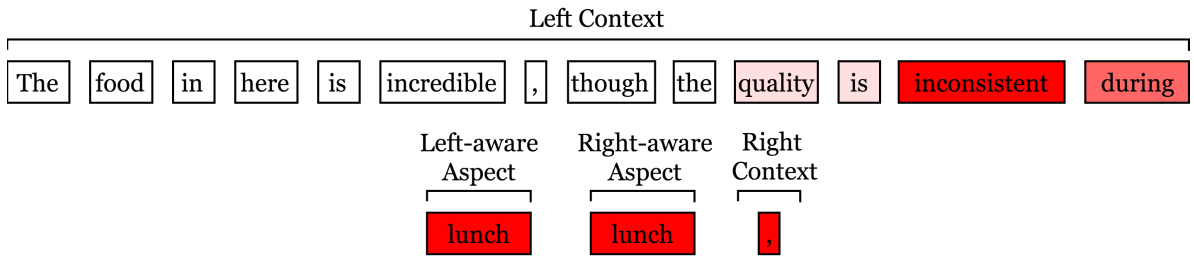
Attention visualizations of the LCR-Rot, LCR-Rot-hop, and CABASC models for the phrase 'The food in here is incredible, though the quality is inconsistent during lunch' for aspect 'lunch':



(a) CABASC attention visualization



(b) LCR-Rot attention visualization



(c) LCR-Rot-hop attention visualization

- Red color indicates word attention (dark color means high attention)
- CABASC and LCR-Rot wrongfully pays attention to the word 'incredible' (which belongs to aspect 'food') [incorrect classification]
- LCR-Rot-hop rightfully gives more attention to 'inconsistent' [correct classification]

Evaluation

The software was implemented in `Python 3` using:

- NLTK for tokenization
- WordNet for lemmatization
- GloVe word embeddings
- TensorFlow for deep learning (using the GPU)
- VADER for sentence sentiment score
- OWLready2 for ontology manipulation and reasoning

Software available at `https://github.com/ofwallaart/HAABSA`

Concluding Remarks

Conclusion

- We have proposed HAABSA, a two-step hybrid approach for sentence-level aspect-based sentiment analysis:
  1. Knowledge-Based Reasoning
  2. Deep Learning:
     - LCR-Rot (Left-Center-Right Separated Neural Network with Rotatory Attention)
     - LCR-Rot-inv (Inverted LCR-Rot)
     - LCR-Rot-hop (Multi-hop LCR-Rot)
- Obtained better results than the state-of-the-art CABASC and LCR-Rot methods
  - Best results for Ont-LCR-Rot-hop (3 hops)

Future Work

- Develop a learning algorithm for the lexicalized domain sentiment ontology (reuse work on ontology learning)
- Propose a solution to deal with implicit aspects (finding target proxies based on word similarity)

References

Qiao Liu, Haibin Zhang, Yifu Zeng, Ziqi Huang, and Zufeng Wu. Content Attention Model for Aspect Based Sentiment Analysis. In *Proceedings of the 2018 World Wide Web Conference (WWW 2018)*, pages 1023–1032. IW3C2, 2018.

Kim Schouten and Flavius Frasincar. Ontology-Driven Sentiment Analysis of Product and Service Aspects. In *Proceedings of the 15th Extended Semantic Web Conference (ESWC 2018)*, volume 10360 of *LNCS*, pages 608–623. Springer International Publishing, 2018.

Shiliang Zheng and Rui Xia. Left-Center-Right Separated Neural Network for Aspect-based Sentiment Analysis with Rotatory Attention, 2018. *arXiv preprint arXiv:1802.00892*.