Multi-Task Mutual Learning for Joint Sentiment Classification and Topic Detection

Lin Gui, Jia Leng, Jiyun Zhou, Ruifeng Xu, Yulan He

Abstract—Recently, advances in neural network approaches have achieved many successes in both sentiment classification and probabilistic topic modeling. On the one hand, latent topics derived from the global context of documents could be helpful in capturing more accurate word semantics and hence could potentially improve the sentiment classification accuracy. On the other hand, the word-level attention vectors obtained during the learning of sentiment classifiers could carry word-level polarity information and can be used to guide the discovery of topics in topic modeling. This paper proposes a multi-task learning framework which jointly learns a sentiment classifier and a topic model by making the word-level latent topic distributions in the topic model to be similar to the word-level attention vectors in sentiment classifiers through mutual learning. Experimental results on the Yelp and IMDB datasets verify the superior performance of the proposed framework over strong baselines on both sentiment classification and topic modeling. The proposed framework also extracts more interpretable topics compared to other conventional topic models and neural topic models.

Index Terms—Multi-Task Learning, Sentiment Analysis, Neural Topic Models.

1 Introduction

In recent years, advances in neural networks (NNs) have achieved many successes in both text classification and probabilistic topic modeling. In NN-based text classifiers, the inputs to the NN architecture are usually the distributed representations of words which are typically learned based on the local context that they occurred. However, the problem with this setup is that each word is only mapped to a single vector representation without considering the actual topic that it is associated with. For example, for the sentence, '*I like java.*', if it appeared in reviews relating to food, then '*java*' most likely refers to '*coffee*'. However, if the sentence is found in reviews about programming languages, then '*Java*' is likely a programming language. This shows that incorporating the latent topic information could better capture word semantics and hence feeding the topic information into NN-based text classifier learning would likely produce more accurate classification results.

For the NN-based generative topic models, the latent topics are learned based on the global context of words (word co-occurrence patterns in a document), as opposed to the local context windows used in word embedding learning. As such, they have a difficulty in capturing the local contextual information such as negations in review documents. Moreover, typically NN-based topic models take a document representation as an input and then pass it through a number of hidden layers before re-constructing the original document representation. One of the hidden layers is usually assumed to capture the latent topic information of the document. In such a setup, however, it is difficult to incorporate word prior sentiment information (e.g., '*excellent*' carries a positive sentiment) into the NN architecture. Nevertheless, as shown in Figure 1, the attention signals produced during sentiment classifier training sometimes reveal word-polarity information, e.g., '*hackneyed*' in Figure 1 carries a strong negative polarity. Hence, the incorporation of attention signals for the learning of neural topic models could help extracting semantically more coherent topics.

Since jointly learning a neural topic model and a sentiment classifier could bring benefits to both tasks, we could explore multi-task learning (MTL). In a typical MTL framework, as shown in Figure 2(a), the lower layers capture the shared representation across tasks, and the task-specific upper layers are stacked to learn task-relevant representations. However, such a framework is not applicable here since the learned latent topic representations in topic models can not be shared directly with word or sentence representations learned in classifiers, due to their different inherent meanings.

We instead propose a new MTL framework based on mutual learning to jointly learn a topic model and a classifier as shown in Figure 2(b). Recall that in a NN-based topic model, the goal is to use the learned topic representations to reconstruct documents. Hence, the weights connecting the latent topic layer and the reconstruction layer essentially represent the topic probabilities for words. That is, each word is associated with a topic distribution. On the other hand, in a NN-based text classifier, we can have an attention layer calculating the contribution of each word (or attention weight) towards the final document representation before feeding it to a softmax layer for classification. As such, each word is associated with an attention vector which somehow indicates the word-level polarity (as shown in Figure 1). The key idea in our proposed MTL framework is to design a mutual learning strategy to maximise the similarity between the latent topic distribution of a word in the NN-based topic model and the attention vector of the corresponding word

Corresponding Author: Yulan.He@warwick.ac.uk

- *L. Gui and Y. He are with the Department of Computer Science, University of Warwick, UK.*
- *L. Jia, J. Zhou and R. Xu are with School of Computer Science, Harbin Institute of Technology, China.*

Manuscript received xx, 2019; revised xx, 2019.

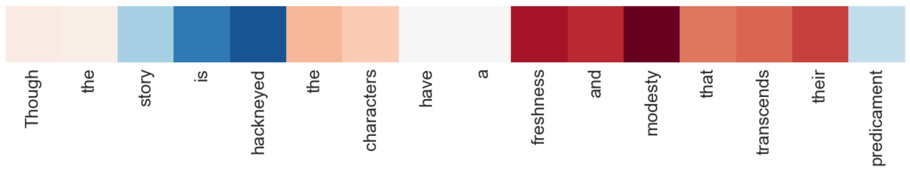


Fig. 1. Attention signals in sentiment classification. Darker blue colour denotes higher attention weights for the negative sentiment while darker red colour denotes higher attention weights for the positive sentiment.

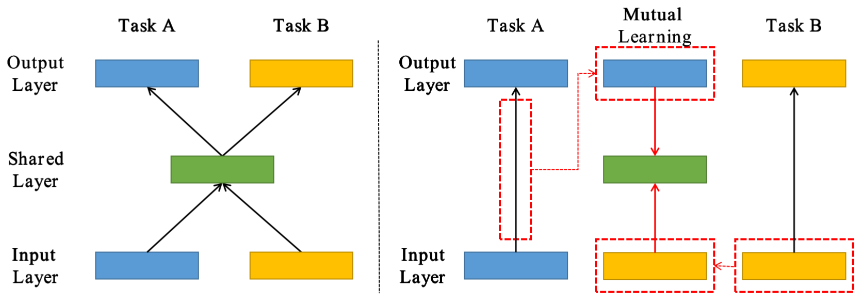


Fig. 2. Two Multi-Task Learning (MTL) architectures: (a) MTL with a shared layer; (b) MTL with mutual learning between weights and units. Here, Task A represents the topic modeling task and Task B represents the sentiment classification task. The red box enclosing the weight links in Task A denotes the topic-word distributions while the red box enclosing the neural network nodes in Task B denotes the word-level attention vectors.

in the NN-based text classifier as illustrated in Figure 2(b).

In order to explain our key idea better, we give an example in Figure 3 to illustrate the word-level attention weights generated from a traditional sentiment classification model and our proposed topic-sentiment mutual learning. We can observe that the traditional sentiment classification model puts a higher attention weight on the polarity word '*incredible*'. But with topic-sentiment mutual learning, higher attention weights are not only put on the polarity word '*incredible*', but also on the associated topic indicated by the words '*acting*' and '*movie*'. We argue that mutual learning would benefit sentiment classification since it enriches the information required for the training of the sentiment classifier (e.g., when the word '*incredible*' is used to describe '*acting*' or '*movie*', the polarity should be positive). At the same time, mutual learning could also benefit topic model learning since it encourages the clustering of words which are not only relevant under a similar topic but are also linked with a similar polarity.

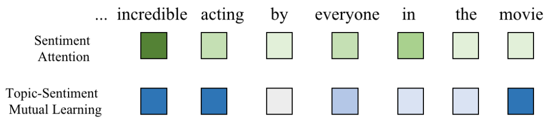


Fig. 3. Different attention weights generate from traditional sentiment classification and our proposed mutual learning. Darker color indicates higher attention weights.

We summmarise the main contributions below[1]:

- we propose a multi-task mutual learning framework for jointly learning a neural generative topic model and a NN-based text classifier;
- We propose a novel mutual learning method to maximise the similarity between the topic distribution of a word in the NN-based topic model and the attention vector of the corresponding word in the NN-based text classifier;
- We evaluate our model on subsets of the Yelp and IMDB datasets and show that our model achieves superior performance compared with several strong baselines on both sentiment classification and topic extraction. On the original full datasets, our proposed method still beats the baselines most of the time on sentiment classification even though our topic modeling component can only be trained with a reduced vocabulary.
- The proposed framework also extracts more semantically coherent and interpretable topics compared to other conventional topic models and neural topic models.

The rest of the paper is organised as follows. Section 2 provides a review of related works on topic modeling, sentiment classification and multi-task learning for NLP. Section 3 presents our proposed multi-task mutual learning framework. Section 4 discusses experimental setup and evaluation results. Finally, Section 5 concludes the paper and outlines the future directions.

2 Related Work

Our work is related to three lines of research: topic modeling, text classification, and multi-task learning for NLP.

2.1 Topic modeling

Probabilistic topic models have been used widely in NLP. Typically, words are assumed to be generated from latent

1. We release our source code at OpenAIRE, https://zenodo.org/record/3731361

topics which can be inferred from data based on word co-occurrence patterns. Topic models are often learned using the Markov chain Monte Carlo method [1], [2] in which model training and inference are approximated using sampling. Many variants of topic models have been proposed to take into account of sentiment [3], [4], [5], author information [6], contextual information [7] and word embeddings [8], [9], [10].

In recent years, the neural network based topic models have been proposed for many NLP tasks, such as information retrieval [11], aspect extraction [12] and sentiment classification [13]. The basic idea is to construct a neural network which aims to approximate the topic-word distribution in probabilistic topic models. Additional constraints, such as incorporating prior distribution [14], enforcing diversity among topics [15] or encouraging topic sparsity [16], have been explored for neural topic model learning and proved effective. However, most of these algorithms take the auto-encoder as the basis learner to fit the distribution function. Due to the drawback in integrals, the generalization ability of the auto-encoder is limited.

More recently, the Variational Auto-Encoder (VAE) [17] has been proved more effective and efficient to approximating deep, complex and underestimated variance in integrals [18], [19]. However, these VAE-based topic models [17], [20], [21], [22] focus on the construction of deep neural networks to approximate the intractable distribution between observed words and latent topics. Intuitively, training the VAE-based supervised neural topic models with class labels [23], [24] may generate better features for classification. However, existing supervised neural topic models treat the class labels as weights which are distributed across words during training. It thus ignores the rich contextual information such as dependencies among words/phrases in a sentence and the ordering of sentences in a document, which is important for many downstream NLP tasks including sentiment analysis and opinion mining.

2.2 Sentiment Classification

Sentiment classification can be solved by supervised statistical learning models [25], [26], [27]. Traditional feature-engineering based sentiment classification models usually focus on extracting efficient features including lexical features [26], topic-based features [3] and ontology-based features [28]. With the rapid development of deep learning, NN based models have shown promising performance on sentiment classification. Various architectures of neural networks including Convolution Neural Network (CNN) [29] and Recurrent Neural Network (RNN) [30] have been proposed for sentiment classification. In document-level sentiment classification, the hierarchical semantic composition of a document can be modeled by hierarchical models such as the Gated Recurrent Neural Network [31] and the Hierarchical Attention Network [32]. More recently, training neural networks with large-scale pre-trained word embeddings gives significantly improvements in text classification [33], [34].

In recent years, neural attention models have been commonly applied to NLP tasks, including neural machine translation [35], text summarisation [36] and textual inference [37], and have also achieved encouraging result in sentiment analysis [38]. Therefore, the most commonly seen sentiment classification models now focus on incorporating the attention mechanism into the NN-based text classification models. Chen et al. [39] used the author and product information to obtain the attention signals with a hierarchical neural network for sentiment classification on product reviews. Tang et al. [40] deployed the memory network with attention for aspect-level sentiment classification. He et al. [13] stacked an auto-encoder layer in the memory network to obtain the aspect-specific attentions to improve the aspect-level sentiment classification. Besides the aforementioned methods, complicated neural module in networks [41], [42], prior knowledge obtained from a sentiment lexicon [43], and author profiles [44] have also been considered in sentiment classification. However, existing neural attention models largely derive the attention weights or signals based on lexical matching between the hidden state representation of the current lexical unit (word or sentence) and a common context vector shared across sentences or documents. They mostly ignore the latent topic information which captures the semantic information in the global context.

2.3 Multi-Task Learning

Multi-task learning (MTL) is an important machine learning mechanism that improves the generalisation performance by learning a task together with other related tasks. It usually has a common layers which learns a shared representation across tasks, then stacks several task-specific upper layers to learn task-relevant representations. MTL has been successfully applied in many NLP tasks including classification [45], [46] and sequence tagging [47], [48], [49], [50]. However, most MTL research focuses on supervised learning [51]. To the best of our knowledge, there is no MTL framework for jointly learning an unsupervised model such as a generative topic model and a supervised model such as a text classifier.

Another challenging is that, most MTL research focuses on relevant tasks. For example, Liu et al. [52] proposed a RNN with shared layers for four tasks, which are all supervised text classification tasks. For sequence tagging [47], MTL is used for chucking, named entity recognition and part-of-speech tagging, which are all related. In this paper, the goal of our research is to build a framework which benefits both the unsupervised neural topic model and the supervised sentiment classifier. To the best of our knowledge, there is no MTL framework for jointly learning an unsupervised model and a supervised model simultaneously in NLP.

3 Proposed Method

In this section, we will introduce our proposed MTL framework with mutual learning for topic detection and sentiment classification. The overall architecture is illustrated in Figure 4. We propose a novel MTL framework based on the following observation. The weights of the decoder in a neural topic model indicate the association probabilities between words and topics, while the attention signals in an attention-based classification model captures the importance of words/sentences contributing to the overall sentiment classification. Thus, if we could make these two distributions as similar as possible, we can potentially generate polarity-bearing topics and at the same time achieve higher sentiment classification results with the topical information incorporated. In the following, we will first briefly introduce the basic neural topic model and the classification model, then present our proposed mutual learning mechanism and the training strategy for MTL.

3.1 Neural Topic Model

Inspired by [18], [19], we assume the document-level topic distribution can be approximated by an Multi-Layer Perceptron (MLP) taking as input a multivariate Gaussian distribution. Then we can build our neural topic model based on VAE, which consists of two main components, the Inference Network and the Generation Network. For the Inference Network, we use VAE to approximate the posterior distribution over topics for all the training instances. In the Generation Network, the words are generated via Gaussian softmax construction from the topic distribution generated by the Inference Network. The architecture of the neural topic model is shown in the left part of Figure 4 and we describe the model in more details below.

Inference Network. Following the idea of VAE which computes a variational approximation to an intractable posterior using MLPs, we define two MLPs f_{μ_θ} and f_{Σ_θ} which takes as input the word counts in a document, $\mu_\theta = f_{\mu_\theta}(\boldsymbol{w}_d)$, $\Sigma_\theta = \text{diag}(f_{\Sigma_\theta}(\boldsymbol{w}_d))$, and outputs mean and variance of a Gaussian distribution, both being vectors in $\mathbb{R}^K$. Here, 'diag' converts a column vector to a diagonal matrix. For a document $\boldsymbol{w}_d$, its variational distribution is $q(\theta) \simeq \mathcal{N}(\mu_\theta, \Sigma_\theta)$. With such a formulation, we can generate samples from $q(\theta)$ by first sampling $\epsilon \sim \mathcal{N}(0, I^2)$ and then computing $\hat{\theta} = \sigma(\mu_\theta + {\Sigma_\theta}^{1/2}\epsilon)$.

Generation Network. We feed the sampled $\hat{\theta}$ to two MLPs to generate $\boldsymbol{z}_d$. Here, $\boldsymbol{z}_d$ is a K-dimensional latent topic representation of document d. The probability of word $w_{d,n}$ can be parameterised by another network,

$$p(w_{d,n}|\boldsymbol{w}, z_d) \propto \exp(\boldsymbol{m}_d + \boldsymbol{W}_t \cdot \boldsymbol{z}_d), \quad (1)$$

where $\boldsymbol{m}_d$ is the V-dimensional background log-frequency word distribution, $\boldsymbol{W}_t \in \mathbb{R}^{V\times K}$.

With the sampled $\hat{\theta}$, for each document $\boldsymbol{w}_d$, we can estimate the Evidence Lower Bound with a Monte Carlo approximation using L independent samples:

$$\mathcal{L}_t(\boldsymbol{w}_d) \approx \frac{1}{L}\sum_{l=1}^{L}\sum_{n=1}^{N_d} \log p(w_{d,n}|\hat{\theta}^{(l)}) - KL(q(\boldsymbol{z}_d|\boldsymbol{w}_d)||p(\boldsymbol{z}_d)), \quad (2)$$

where the first term is given by Eq. (1), and the second term is:

$$\begin{aligned} &- KL(q(\boldsymbol{z}_d|\boldsymbol{w}_d)||p(\boldsymbol{z}_d)) \\ =&\frac{1}{2}\sum_{k=1}^{K}(1 + \log((\sigma_{\theta,k})^2) - (\mu_{\theta,k})^2 - (\sigma_{\theta,k})^2), \end{aligned} \quad (3)$$

where K is the number of topics.

At the reconstruction layer, we stack a single-layer network to capture the sampling weights between each words and the latent topics.

3.2 Neural Text Classification Model

For the text classification model, we use a hierarchical RNN to model a document. Assuming that a document $\boldsymbol{w}_d$ contains M_d sentences, $\boldsymbol{w}_d = \{s_1, s_2, ...s_{M_d}\}$, and the word embedding of j-th word in i-th sentence is w_i^j. Then, the representation of sentence s_i can be obtained by the following steps:

$$\begin{gathered} x_i^j = W \cdot w_i^j, \\ \overrightarrow{h_i^j} = \overrightarrow{\text{GRU}}(x_i^j), \quad \overleftarrow{h_i^j} = \overleftarrow{\text{GRU}}(x_i^j), \quad h_i^j = \overrightarrow{h_i^j} \oplus \overleftarrow{h_i^j}, \\ u_i^j = \tanh(W_w \cdot h_i^j + b_w), \\ \alpha_i^j = \frac{\exp(u^T \cdot u_i^j)}{\sum_t \exp(u^T \cdot u_i^t)}, \quad s_i = \sum_{j=1}^{n} \alpha_i^j \cdot h_i^j, \end{gathered} \quad (4)$$

where $\overrightarrow{\text{GRU}}$ and $\overleftarrow{\text{GRU}}$ are bi-directional gated recurrent neural units for RNN, W, W_w, b_w are learned parameters in the classification model, and u_i^j is the attention vector of j-th word in i-th sentence, α_i^j is the attention signal captured by u_i^j, s_i is the learned representation for the i-th sentence in document $\boldsymbol{w}_d$. Then, we can learn the representation of document $\boldsymbol{w}_d$ with the similar architecture taking the input as a sequence of sentence representations. Finally, a softmax layer is stacked at the top to predict the class labels of documents by cross entropy loss between the predicted labels and the true labels.

$$\mathcal{L}_c(\boldsymbol{w}_d) = -\sum p \cdot \log[\text{softmax}(W_d \cdot \boldsymbol{w}_d + b_d)], \quad (5)$$

where the output of $\text{softmax}(W_d \cdot \boldsymbol{w}_d + b_d)$ is the distribution of predicted labels and p is the distribution of true labels.

3.3 Two Strategies for Multi-Task Learning

Before presenting our proposed mutual learning strategy for MTL, we first describe a simple setup, termed as '*Hard Attentions*', by forcing the topic distribution and the attention vector of a word to be the same by taking the mean value of these two vectors.

3.3.1 *Multi-Task Learning with Hard Attentions*

In the hard attention mechanism, we build a lookup table for each word in the vocabulary. The vector which is associated with a word is the mean value of its word attention vector obtained from sentiment classifier training and the decoder weights for topic generation obtained from the neural topic model.

Formally, for $W_t \in \mathbb{R}^{V\times K}$ in Eq. (1), let ${w_{ij}}^2$ be an entry in W_t, which stands for the generative probability of the i-th word from the j-th topic, then the latent topic distribution for the i-th word can be represented as $w_i' = \{w_{i1}, w_{i2}, ..., w_{iK}\}$, where K is the total number of topics. We can also obtain the attention vector u_i' for the i-th word from Eq. (4), $u_i' = \tanh(W_w \cdot h_i + b_w)$. Here, h_i

2. Note that we abuse the notation of w_{ij} here. Its meaning should be clear from the context.

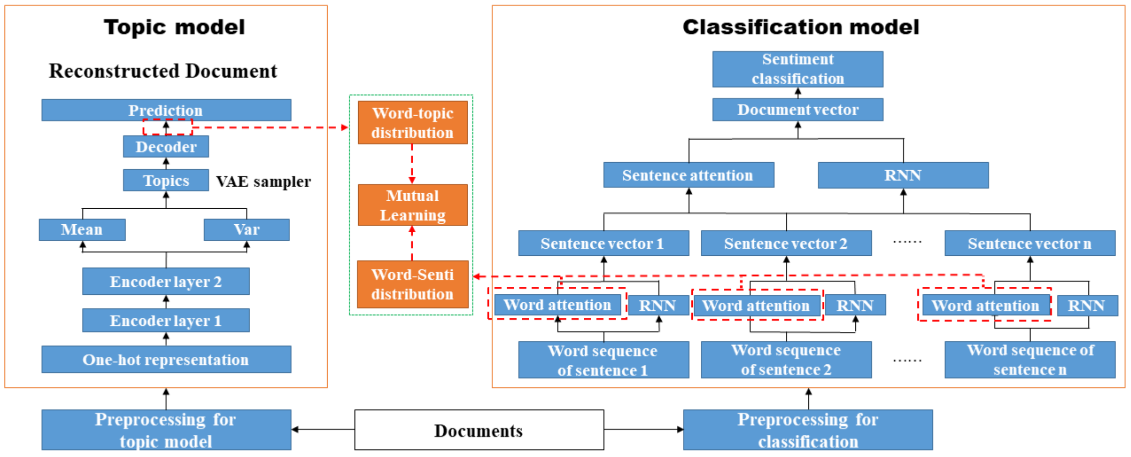


Fig. 4. Our proposed MTL with mutual learning for sentiment classification and topic detection.

Algorithm 1 Multi-task Learning with Hard Attention

```
Require: Documents with labels $\{\boldsymbol{w}_d, y_d\}, d = \{1, 2, ..., D\}$, pre-trained word embeddings, the maximum training iterations $T$.
Ensure: Trained topic model and sentiment classifier
1: Initialise model parameters
2: for $j = 1$ to $T$ or until convergence do
3:     Minimise the loss function $\mathcal{L}_{final}$ in Eq.(7)
4:     Update the lookup table by Eq. (6) for each word
5:     Update the weights in the topic decoder and the attention vectors according to the lookup table
6: end for
7: Fine tune the model by minimising the task specific loss function for each task
```

is the hidden representation of the i-th word learned by GRU. In order to reshape the attention vector and topic distribution to the same size, we fix the dimension of the attention vectors as K. For the i-th word in the lookup table, the vector is:

$$V_i = \frac{u_i' + w_i'}{2} \tag{6}$$

During training, we take topic modeling as task A and sentiment classification as task B. Then, the loss function is:

$$\mathcal{L}_{final} = \sum_d \alpha \cdot \mathcal{L}_t(\boldsymbol{w}_d) + \beta \cdot \mathcal{L}_c(\boldsymbol{w}_d) \tag{7}$$

The lookup table is updated after each epoch of training. Then the weights of the topic decoder and the attention vectors are updated according to the lookup table. The pseudo-code of the training procedure is shown in Algorithm 1.

3.3.2 *Multi-Task Mutual Learning*

In the hard attention model, the neural topic model and the sentiment classifier are essentially trained separately except that the weights of the topic decoder and the attention vectors are forced to be the same by taking the average of their corresponding word-level distributions at the end of each training iteration. Such an ad-hoc update does not take into account the objective function during the decoder weight or attention signal update.

In this subsection, we instead propose a training strategy based on mutual learning. The key idea is to use latent topic distribution of each word obtained from the neural topic model to guide the calculation of word-level attention signals in the text classification model, which essentially incorporates the topic information into the classifier training. On the other hand, the word-level attention vector which potentially carries the word-level polarity information could be used to guide the learning of latent topic distributions in the neural topic model.

The latent topic distribution for each word can be obtained by using the weights connecting the penultimate layer and the reconstruction layer in the neural topic model (as shown in the left part of Figure 4). The attention vector for each word in RNN is stored in u_i^j as in Eq. (4). Mutual learning is used to make the latent topic distribution of a word to be similar to the attention vector of the same word from RNN. The benefits of using such a strategy are: 1) learning latent topics with word-level polarity information derived from classifier training without the need of using any external sentiment lexicons; 2) incorporating the latent topic information into classifier training to improve the classification performance.

Using the same notation introduced in Section 3.3.1 that the latent topic distribution for the i-th word is represented as $w_i' = \{w_{i1}, w_{i2}, ..., w_{iK}\}$, where K is the total number of topics, and the attention vector u_i' for the i-th word is obtained from Eq. (4). We measure and maximise the similarity between the latent topic distribution of the ith word, w_i', and its attention vector, u_i', during the training. Here, we explore four different similarity measurement metrics:

$$\begin{aligned}
\mathcal{O}_1 &= \sum_i \frac{|w_i' \cdot u_i'|}{\|w_i'\|_2 \cdot \|u_i'\|_2} \\
\mathcal{O}_2 &= \sum_i |w_i' \cdot u_i'| - \|w_i'\|_2 - \|u_i'\|_2 \\
\mathcal{O}_3 &= \sum_i \frac{2}{2 + D_{KL}(w_i'|u_i') + D_{KL}(u_i'|w_i')} \\
\mathcal{O}_4 &= \sum_i \frac{1}{1 + \frac{1}{\frac{1}{D_{KL}(w_i'|u_i')} + \frac{1}{D_{KL}(u_i'|w_i')}}}
\end{aligned} \tag{8}$$

Algorithm 2 Multi-task Mutual Learning

```
Require: Documents with labels $\{\boldsymbol{w}_d, y_d\}, d = \{1, 2, ..., D\}$, pre-trained word embeddings, candidate word vocabulary $\mathcal{V} = \{w'_1, w'_2, ...w'_{\|\mathcal{V}\|}\}$, the maximum training iterations $T$.
Ensure: Trained topic model and classifier
1: Initialise model parameters
2: for $j = 1$ to $T$ or until convergence do
3:     for each mini batch of training instances do
4:         Minimise the loss function $\mathcal{L}_{final}$ in Eq. (7)
5:     end for
6:     for $t = 1$ to $\|\mathcal{V}\|$ do
7:         Optimise the object function $\mathcal{O}$ in Eq. (8) for each $w'_t$
8:     end for
9: end for
10: Fine tune the model by minimising the task specific loss function for each task
```

Here, $\mathcal{O}_1$ is based on cosine similarity measurement, $\mathcal{O}_2$ is an inner product with a regularisation term, $\mathcal{O}_3$ and $\mathcal{O}_4$ are two similarity functions introduced by arithmetic and harmonic mean of KL-divergence, respectively. In order to make w'_i and u'_i comparable, we set the dimension of the attention vector of u'_i to be the same as the number of latent topics. With the similarity function define above, we introduce an additional objective function based solely on the similarity function. The optimal object now is to minimise the loss function defined in Eq.(9):

$$\underset{\theta, w'_i, u'_i \in \mathcal{V}}{\arg\min} \sum_d (\alpha \cdot \mathcal{L}_t(\boldsymbol{w}_d) + \beta \cdot \mathcal{L}_c(\boldsymbol{w}_d)) - \sum_{\mathcal{V}} (\mathcal{O}_m(w'_i, u'_i))) \tag{9}$$

where θ is the set of parameters in the neural topic model and the neural text classification model, $1 \leq m \leq 4$, which stands for four different similarity measurements, respectively.

Note that it is highly ineffective to optimise the loss function defined in Eq. (9) directly. Because in topic model learning, usually text pre-processing is required to remove stop words, rare words and some high frequency words in order to alleviate the data sparsity issue. As such, the vocabulary size would be dramatically reduced. However, in RNN-based classification model learning, the original full vocabulary is usually kept since RNN needs to capture the word sequence information. Hence, for the u'_i in the object function, we only consider the words which occur in the filtered vocabulary for topic modeling to ensure the symmetrical measure in the vector space constructed by our neural networks. In our actual implementation, we optimise the parameter in Eq. (7) and Eq. (8) separately. In each training epoch, we optimise the parameter in Eq. (7) iteratively for each minibatch of training instances to obtain topic and sentiment representations. Then, we update O_m defined in Eq. (8) at the end of the current training epoch. That is, Eq. (7) is optimised more frequently compared to Eq. (8). The pseudo-code of the training procedure is shown in Algorithm 2.

For those words kept in the vocabulary for topic modeling, we can fine-tune the two distributions by making them similar (using the hard attention or mutual learning as proposed in our work). In this way, we can adjust the word representations in both embedding spaces in order to incorporate the sentiment information induced by the classifier and the topic information generated by the topic model. The isometrically embedding learning during classifier training can also ensure the influence of topic distributions to be spread to words which are filtered in topic modeling.

4 EXPERIMENTS

We evaluated our proposed framework on the IMDB and the Yelp datasets[3] . Due to the computational constraints of training topic models, we sample part of the data to ensure balanced class distributions and a suitable size for topic model. Nevertheless, in order to compare with other sentiment classification baselines, we also evaluate our proposed method on the full dataset. The details of the sampled datasets and original dataset are shown in Table 1.

Data	#Class	#docs	Vocab. Size
IMDB	10	348,415	115,831
Yelp2013	5	335,018	211,245
Yelp2014	5	1,125,457	476,191
Yelp2015	5	1,569,264	612,636
Sampled IMDB	10	15,000	55,819
Sampled Yelp	5	39,923	53,823

TABLE 1
Statistics of full datasets.

4.1 Experimental setup

For pre-processing, we performed stop word removal using the mallet toolkit[4] and filtered the most frequent words falling into the upper quantile of 0.025 in the vocabulary. We then selected the most frequent 2,000, 3,000 and 5,000 words to form our vocabulary for the neural topic model.

For classifier training, we keep the full vocabulary since our classifier is built on the hierarchical RNNs. When maximising the similarity function in Eq. (8), we only consider the attention vectors for words which can be found in the vocabulary of the topic model. We set aside 10% training samples as the validation set for parameter tuning.

During training, we use Adam to optimise the parameters in the neural networks. The learning rate is 0.0001, the number of latent topics is set to $[30, 50, 80]$, the mini-batch size is 20, α is 0.1, β is 0.9, and the pre-trained word embeddings for text classification are obtained from word2vec[5] [53] trained from the Google news corpus and the dimension of word vector is 300.

4.2 Sentiment classification

For sentiment classification, we compare our approach with the following baselines.[6]

3. http://ir.hit.edu.cn/~dytang/paper/emnlp2015/emnlp-2015-data.7z
4. http://mallet.cs.umass.edu/import-stoplist.php
5. https://code.google.com/archive/p/word2vec/
6. Whenever the original implementation is available, the link to the source code is given in the footnote. For those methods which were re-implemented by us, they were marked with †.

Method	Accuracy					
	Yelp			**IMDB**		
LightGBM	0.544			0.177		
NGTM$_l$	0.248			0.118		
JST	0.411			0.243		
NTSM	0.392			0.198		
CNN	0.572			0.196		
CNN+	0.580			0.195		
RNN	0.577			0.196		
RCNN	0.594			0.196		
NSC	0.616			0.355		
MTL-RNN	0.610			0.327		
$V = 2,000$	$K = 30$	$K = 50$	$K = 80$	$K = 30$	$K = 50$	$K = 80$
MTL-HA	0.608	0.611	0.599	0.381	0.371	0.378
MTL-ML$_1$	0.610	0.618	0.608	**0.407***	**0.407***	0.396
MTL-ML$_2$	0.610	0.614	0.610	0.406	0.400	0.398
MTL-ML$_3$	**0.621***	**0.628***	**0.621***	0.394	0.399	0.358
MTL-ML$_4$	0.619	0.627	0.619	0.406	0.401	**0.403***
$V = 3,000$	$K = 30$	$K = 50$	$K = 80$	$K = 30$	$K = 50$	$K = 80$
MTL-HA	0.601	0.603	0.596	0.366	0.374	0.387
MTL-ML$_1$	0.611	0.605	0.619	**0.408***	0.402	**0.402***
MTL-ML$_2$	0.612	0.611	0.612	0.396	**0.403***	0.394
MTL-ML$_3$	**0.623***	**0.622***	**0.627***	0.367	0.399	0.376
MTL-ML$_4$	0.620	0.617	0.619	0.399	0.401	0.397
$V = 5,000$	$K = 30$	$K = 50$	$K = 80$	$K = 30$	$K = 50$	$K = 80$
MTL-HA	0.601	0.597	0.583	0.377	0.362	0.381
MTL-ML$_1$	0.616	0.608	0.618	0.402	0.402	0.395
MTL-ML$_2$	0.597	0.617	0.605	0.371	0.396	0.396
MTL-ML$_3$	**0.621***	0.617	**0.628***	0.395	**0.411***	0.385
MTL-ML$_4$	0.619	**0.618***	0.621	**0.406***	0.398	**0.408***

TABLE 2
Sentiment classification accuracy on sampled dataset. (* denotes proposed best method vs. the best in reference methods using t-test with $p < 0.01$, V is the size of vocabulary and K is the number of topics.)

- LightGBM[7]: The Gradient Boosting Model (GBM) [54] with n-gram features. Here, the n-gram features include the universal set of uni-grams, the top 500 most frequent bi-grams and tri-grams;
- NGTM$_l$[8]: Neural Generative Topic Model (NGTM) with sentiment labels [19]. Here, the basic classifier we choos is the Gradient Boosting Model (GBM) [54];
- JST[9]: Joint Sentiment-Topic Model [3], [4], a Bayesian modeling approach which jointly models sentiments and topics. The model is initialised with the word prior polarity derived from the MPQA sentiment lexicon [55]. The number of latent topics is 100 and the size of vocabulary is 5,000;
- NTSM: A Neural Sentiment Topic Model [23] with separate topic and sentiment representation. The prior knowledge of sentiment is from the MPQA sentiment lexicon [55] and the sentiment label of documents. The number of latent topics is 100 and the size of vocabulary is 5,000;
- CNN[10]: Convolutional neural network [29], which is a strong baseline and widely used in sentiment classification [56]. The kernel sizes are $\{2, 3, 4\}$, and the number of kernels of each size is 50;
- CNN+†: Incorporating n-gram features into the pooling layer in CNN for sentiment classification;
- RNN†: Recurrent neural network for sentiment classification [30]. The size of hidden states is 150;
- RCNN[11]: A hierarchical neural network stacked with CNN and RNN layers for sentiment classification [31]. The convolutional kernel sizes are $\{2, 3, 4\}$, the number of kernels of each size is 50, and the size of hidden states in RNN is 150;
- NSC: A neural sentiment classification method [39] with hierarchical RNNs and an attention mechanism [32]. We use the re-implementation from [39][12] with the default parameter settings.

For our proposed framework, we consider the following variants:

- MTL-RNN: Using RNN to reconstruct document instead of VAE in our proposed MTL framework.
- MTL-HA: Our MTL framework without mutual learning. Instead, it simply uses the average of word attentions and topic weights as the '*hard-attention*' to allow the sharing of the information between the two networks.
- MTL-ML: Our proposed MTL with mutual learning with four different similarity measures, each one marked as MTL-ML$_m$, $m = 1, 2, 3, 4$.

The results of sentiment classification are shown in Table 2. LightGBM is a decision tree based classification model. We train it with n-gram or topic features. Its performance is the worst compared to NN-based classification methods. RCNN and NSC are based on the hierarchical learning architecture. The improvements of these two methods over

7. https://scikit-learn.org/
8. https://github.com/ysmiao/nvdm
9. https://github.com/linron84/JST
10. https://github.com/yoonkim/CNN$_sentence$
11. http://ir.hit.edu.cn/ dytang/paper/emnlp2015/codes.zip
12. https://github.com/thunlp/NSC

Method	Yelp						IMDB					
	PPL	C_v	**PPL**	C_v	**PPL**	C_v	**PPL**	C_v	**PPL**	C_v	**PPL**	C_v
$V = 2,000$	$K = 30$		$K = 50$		$K = 80$		$K = 30$		$K = 50$		$K = 80$	
JST	1375.5	0.498	1438.8	0.498	1312.5	0.491	2573.8	0.517	2900.8	0.493	2348.3	0.502
NVDM	1980.1	0.492	1976.0	0.480	1985.9	0.536	1990.3	0.513	1872.3	0.517	1980.8	0.524
NGTM	1843.1	0.505	1851.9	0.511	1991.5	0.522	2064.2	0.493	2083.1	0.565	2038.2	0.505
NGTM$_l$	1901.4	0.506	1733.9	0.514	2095.9	0.480	1887.8	0.485	2083.7	0.497	2719.3	0.535
NTSM	1729.8	0.505	1918.5	0.495	1807.6	0.507	1796.1	0.512	1829.5	0.531	2131.4	0.527
MTL-HA	1218.7	0.499	1243.7	0.501	1258.7	0.489	861.4	0.523	880.3	0.521	881.6	0.519
MTL-ML$_1$	1299.4	0.481	1235.1	**0.524***	1236.5	0.541	917.1	0.492	863.3	0.565	862.7	0.556
MTL-ML$_2$	1301.9	0.483	1234.5	0.506	1234.5	**0.553***	914.6	0.501	863.0	0.555	859.6	**0.598***
MTL-ML$_3$	1219.5	0.516	1219.9	0.473	1233.7	0.491	**857.7***	**0.537***	864.7	0.473	862.1	0.496
MTL-ML$_4$	**1216.6***	**0.521***	**1183.9***	0.504	**1190.9***	0.497	860.9	0.531	**841.6***	**0.571***	**839.5***	0.492
$V = 3,000$	$K = 30$		$K = 50$		$K = 80$		$K = 30$		$K = 50$		$K = 80$	
JST	2178.9	0.561	2144.7	0.522	2298.5	0.542	2988.1	0.528	3137.7	0.579	2897.1	0.533
NVDM	2558.8	0.543	2877.8	0.500	2675.9	0.514	2256.9	0.521	2936.6	0.551	2471.7	0.520
NGTM	2619.2	0.511	2840.1	0.488	2927.7	0.525	2300.1	0.531	2772.9	0.586	2454.6	0.508
NGTM$_l$	3253.2	0.480	2650.7	0.488	4329.5	0.519	3599.4	0.533	3321.6	0.586	5080.5	0.521
NTSM	2510.9	0.497	2571.3	0.484	2999.0	0.484	2381.6	0.580	2698.8	0.521	3046.4	0.540
MTL-HA	1801.1	0.551	1788.4	0.521	1787.6	0.544	1289.1	0.561	1261.4	0.537	1211.7	0.559
MTL-ML$_1$	1755.1	0.554	1764.9	0.538	1754.9	0.578	1201.8	0.532	1203.1	0.588	1200.5	0.578
MTL-ML$_2$	1755.0	0.549	1762.1	0.480	1753.8	0.505	1199.6	0.556	1206.1	**0.604***	1197.3	0.467
MTL-ML$_3$	1751.5	0.579	1749.3	0.464	1752.0	0.500	1198.3	0.543	1133.4	0.488	1199.3	0.550
MTL-ML$_4$	**1750.5***	**0.582***	**1706.9***	**0.595***	**1749.4***	**0.584***	**1194.8***	**0.598***	**1131.3***	0.489	**1191.9***	**0.602***
$V = 5,000$	$K = 30$		$K = 50$		$K = 80$		$K = 30$		$K = 50$		$K = 80$	
JST	2171.4	0.546	3451.1	0.501	3133.4	0.551	3298.0	0.521	3361.7	0.516	3470.9	0.535
NVDM	2558.8	0.540	4576.9	0.523	4369.5	0.501	3479.0	0.581	4833.6	0.516	3751.7	0.533
NGTM	2619.3	0.522	4072.1	0.526	5112.8	0.525	3463.0	**0.627**	4593.3	0.564	3805.1	0.549
NGTM$_l$	3253.2	0.515	4500.6	0.513	7636.2	0.498	5929.6	**0.627**	5539.6	0.516	7798.5	0.549
NTSM	2510.9	0.519	3972.4	0.517	4181.6	0.496	4442.1	0.566	4045.8	0.522	4521.5	0.522
MTL-HA	2688.1	0.551	2651.7	0.509	2541.7	0.513	1801.9	0.519	1847.7	0.523	1801.4	0.499
MTL-ML$_1$	2664.8	0.536	2620.8	0.541	2665.1	0.562	1799.9	0.541	1809.8	0.587	1797.0	0.419
MTL-ML$_2$	2664.0	0.476	2614.9	0.462	2663.3	0.434	1796.7	0.520	1816.5	**0.599***	1793.1	0.485
MTL-ML$_3$	2662.5	0.491	**2537.6***	0.479	2656.3	0.500	1791.2	0.574	**1660.8***	0.480	1791.8	0.499
MTL-ML$_4$	**2652.8***	**0.607***	2540.8	**0.543***	**2654.1***	**0.588***	**1788.6***	0.574	1699.7	0.531	**1775.7***	**0.574***

TABLE 3
Perplexity and topic coherence results on sampled dataset.(* denotes proposed best method VS. the best in reference methods using Mc- Nemars test $p < 0.01$, V is the size of vocabulary and K is the number of topics.)

other baselines show the effectiveness of hierarchical learning. With the attention mechanism, NSC shows further improvements over RCNN.

In our proposed framework, the sentiment classifier shares attention weights with topic-word distribution obtained from the neural topic model. We would like to find out if the vocabulary size affects the performance of sentiment classification. In another set of experiments, we evaluate the impact of different vocabulary size {2k, 3k and 5k} on the sentiment classification results.

Among our proposed variants, MTL-HA simply used the '*hard-attention*' to ensure the sharing between the latent topic distributions and the word-level attention vectors without mutual learning. Interestingly, MTL-HA outperforms most baselines. Nevertheless, it performs worse compared to MTL-ML$_m$ ($m = 1, 2, 3, 4$), which are based on mutual learning. MTL-ML$_3$ and MTL-ML$_4$, which are based on KL-divergence, achieve better performance than MTL-ML$_1$ and MTL-ML$_2$ on the Yelp dataset. MTL-ML$_1$ and MTL-ML$_2$, which use cosine similarity based measures, slightly outperform MTL-ML$_3$ and MTL-ML$_4$ on the IMDB dataset when the vocabulary size is 2k and 3k. For the vocabulary size of 5k, MTL-ML$_3$ and MTL-ML$_4$ give the best results on both datasets.

By cross comparing the results with different vocabulary sizes, we notice that the best result is achieved by MTL-ML$_3$ when the vocabulary size is set to 2k on the Yelp dataset. Increasing the vocabulary size results in slightly decreased classification accuracy. For the IMDB dataset, the sentiment classification performance is less sensitive to the vocabulary size. Overall, varying the vocabulary size does not impact the sentiment classification performance much.

We also use RNN instead of VAE to reconstruct the documents in MTL (denoted as MTL-RNN in Table 2). However, the performance is slightly lower than NSC. One possible reason is that using RNN to reconstruct documents is essentially the same as training a language model on the training set. Since we use word embeddings pre-trained on a large-scale data as input to MTL-RNN, it did not gain much on the current training set.

In conclusion, compared with the best baseline model, NSC, our proposed models give the improvement in accuracy ranging between 1.2% and 5.6%.

4.3 Topic modeling

For topic extraction, we compare our approach with the following baselines:

- JST[13]: Joint Sentiment-Topic Model [3], [4], a Bayesian modeling approach which jointly models sentiments and topics. The model is initialised with the word prior polarity derived from the MPQA sentiment lexicon [55];

13. https://github.com/linron84/JST

- NVDM[14]: Neural Variational Document model [18] which is based on VAE for topic extraction from text;
- NGTM†: Neural Generative Topic model [19], also based on VAE, but additionally incorporating the log background frequency of words to better deal with sparsity;
- NGTM$_l$: NGTM trained with document-level sentiment labels.
- NTSM†: A Neural Sentiment Topic Model [23] with separate topic and sentiment representation. The prior knowledge of sentiment is from the MPQA sentiment lexicon [55] and the sentiment label of documents.

The models are evaluated based on perplexity (PPL, lower is better) and topic coherence (C_v measures) [57] (higher is better). The results are shown in Table 3. JST is a conventional Bayesian model with parameters estimated by Gibbs sampling, while NVDM and NGTM are neural topic models. JST and NTSM utilizes the MPQA lexicon to obtain the word-level prior sentiment information and NGTM$_l$ is trained with the document-level sentiment class labels.

It can be observed that among the baselines, the conventional topic model JST outperforms the others on perplexity measures on the Yelp dataset. But on the IMDB dataset, the results are mixed. Neural topic models such as NVDM, NGTM, and NTSM perform better than the others with 2k or 3k vocabulary size, but perform worse than JST with 5k vocabulary size. In terms of topic coherence, NGTM gives the best overall results compared to other baselines. Incorporating supervised sentiment label information (NGTM$_l$, NTSM) does not boost the topic coherence measures.

Among our proposed MTL variants, MTL-ML$_3$ and MTL-ML$_4$ give lower perplexity across different vocabulary sizes on both the Yelp and IMBD datasets. It shows that using similarity measures based on the KL divergence is more effective compared to those based on cosine similarity measures on perplexity. For the coherence scores, MTL-ML$_4$ perform better on the Yelp dataset in most cases when the vocabulary size grows. The MTL-ML$_2$ works well when the topic number is 50.

When compared with the baselines, we can observe that our proposed MTL frameworks give superior performance by a large margin in perplexity on both the Yelp and IMBD datasets. For topic coherence, both MTL-ML$_3$ and MTL-ML$_4$ outperform all the baselines on both datasets with 2k vocabulary size. Increasing the vocabulary size, MTL-ML$_4$ gives the best results on the Yelp dataset, while MTL-ML$_2$ beats all the other models on the IMDB dataset when the topic number is 50.

We also compare the models with the varying topic number and observed generally improved perplexity results with the increasing number of topics. However, the topic coherence values fluctuate across topic numbers. Overall, we can conclude that the best perplexity results on both datasets are obtained by MTL-ML$_4$ with the vocabulary size of 2k and topic number 30. The best topic coherence results are achieved using MTL-ML$_4$ on Yelp and MTL-ML$_2$ on IMDB.

[14]. https://github.com/ysmiao/nvdm

Topic	Top Words from NGTM in IMDB
1	hitler 1940 holden lands lethal morricone repulsive deserving
2	chainsaw offended undead marc ghostbusters deniro smiling realistically
3	witches shake scariest rats patch psychiatric kong cartoons
4	dumbest boring hopelessly insipid hack dismal disastrous nonsensical
5	razzie executives marilyn masses rousing abysmal records glance
Topic	**Top Words from MTL-ML$_4$ in IMDB**
1	horrendous messy solely holy inexplicably avoided ambiguous inferior
2	amazed expert greeting saluted grace model drunken beauty
3	humbert pleasantville joanna caesar damien wolverine mick boogeyman
4	witnesses careers apocalypse teams responsibility viewings panic freak
5	ship flame sorbet delighted moms suns mellow halibut
Topic	**Top Words from NGTM in Yelp**
1	ricotta sweets romaine aioli masala tikka nutella gras provolone
2	paste connoisseur hashbrowns croissant thicker garnish powder tastier
3	disgusted questioned ignoring flagged false lied insult
4	unhelpful lied scam ringing false reply argued loudly
5	pics ad tex aunt walnuts oddly responsive mash
Topic	**Top Words from MTL-ML$_4$ in Yelp**
1	letdown miserably mediocrity frustrating horrendous sadness disastrous sloppy
2	exceptionally unbelievably superbly solely blatant amazed unintentional astonishing
3	hockey player button jersey hipsters questionable incorrectly brew league
4	amc pics elevator buzz lane spotted battered colombian patience
5	crusty barbecue diced kungpao quesadillas chunky pastas hashbrowns

TABLE 4
Example topic words from NGTM and MTL-ML$_2$ (Vocabulary size = 2,000, topic = 50, the sentiment words are highlighted with color, red for negative and green for positive).

We show example topics extracted from IMDB and Yelp by NGTM and MTL-ML$_4$ in Table 4 when the vocabulary size is set to 2k and topic number is 50. In IMDB, the topics extracted by NGTM are more relating to different categories of movies. For example, Topic 1 is about World War II, Topic 2 is about a thriller and Topic 3 contains words relating to cartoons. There are also some opinion words, which can be found in topic 4 and 5. In topic 5, the NGTM mixes positive and negative sentiment words into a same topic. It shows that the topic model lacks the capability to identify sentiment polarity.

The topic results produced by MTL-ML$_4$ contain semantic information from the embedding space. For example, Topic 1 contains largely negative adjective words and topic 2 contains many positive words. Interestingly, Topic 3 contains names of actors or movie characters. One possible reason is that people names might share some contextual patterns. As such, their hidden states in RNN tend to be mapped to similar attention vectors. Due to mutual learning, the attention vectors will influence the topic distributions so as to form a topic of names. Similar to NGTM, there are also some topics about the themes of movies, such as Topic 4 which is likely about a disaster or a catastrophe, and Topic 5 is about holiday and vocations.

On the Yelp dataset, since most reviews are about restaurants, the topics extracted by NGTM are largely relating to cuisines such as Topic 1 and Topic 2, or complaints about services such as Topic 3 and Topic 4. We also notice that most opinion words extracted by NGTM are negative. Topic 5 seems mixing the words about food and service.

On the contrary, our proposed MTL-ML$_4$ can capture not only the negative, but also the positive opinion words, as shown in Topic 1 and Topic 2, respectively, because of the joint learning of topics and sentiment attentions. Another advantage of mutual learning is that the semantic information encoded in the RNN enhances the capability of the neural topic model in capturing the low frequent words. Even the Yelp dataset contains mostly the restaurant reviews, MTL-ML$_4$ extracts topics relating to spot bar or cinema, as shown in Topic 3 and Topic 4, respectively. One possible reason is that the low frequent words are embedded by RNN according to the context, and the embedding space will lead the topic model to generate those words based on the similarity measure during mutual learning.

4.4 Parameter analysis

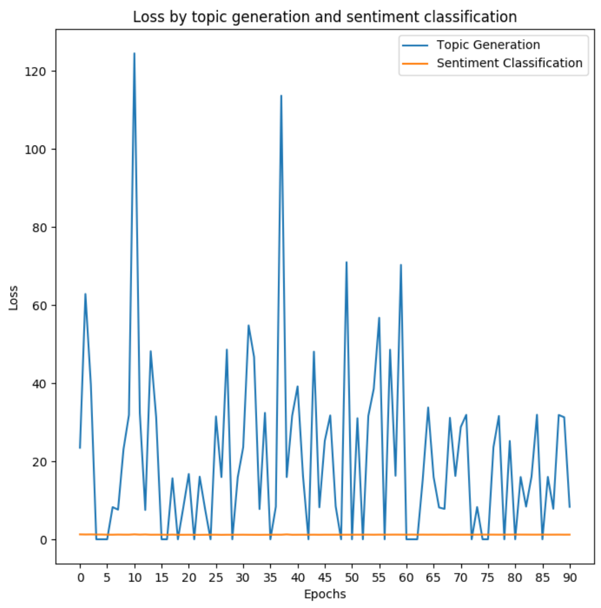
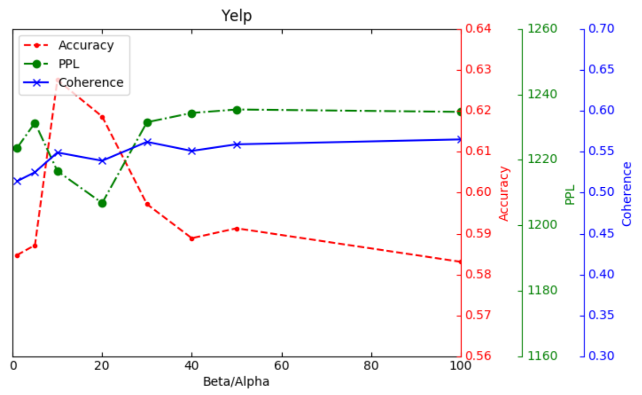


Fig. 5. The output of loss function in topic generation and sentiment classification during the model training.

In our proposed MTL-ML methods, most parameters can be fine-tuned by the validation set. The only parameter requires manual setup is α and β in Eq.(7), which stands for the weights of the two loss functions corresponding to two tasks in multi-task learning. Intuitively, the weights of the two loss functions should be balanced to ensure their equal contributions to the final loss functions. In neural topic model learning, the loss function of topic generation is based on ELBO, which is usually a large number, while for sentiment classifier training, the loss function is based on cross entropy, which is between 0 and 1. To illustrate this, we show in Figure 5 the loss values of two tasks during the model training on the Yelp dataset.

It can be observed that the loss values of the sentiment classifier only fluctuate within a small range while the loss values of topic models have a larger scale. The reason is that although the ELBO can approximate the generative loss theoretically, the generative loss is in fact unbounded [58]. To address this issue, we assign a small value to α, and take $\beta = 1 - \alpha$.



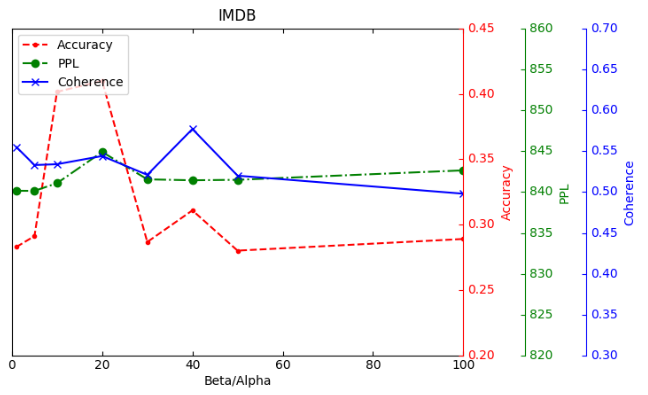


Fig. 6. The performance of sentiment classification and topic generation by proposed MTL-ML$_4$ with different α & β values.

The performances of sentiment classification and topic generation with different ratio of β/α are shown in Figure 6. In this set of experiments, the vocabulary size is 2,000 and the topic number is 50. We can observe that the performance of topic generation (perplexity and topic coherence) is less sensitive to the setting of β/α, while the sentiment classification accuracy is impacted heavily by different values of β/α. In Fig. 6, $\alpha = \{0.05, 0.1\}$ gives the best results on sentiment classification without much degradation of topic PPL and coherence. Hence, we set α to 0.1 in our experiments.

We also explore the influence of the training data size on our model performance. The vocabulary size is 2,000 and the topic number is 50. Results are shown in Table 5. In general, the more the training data used, the better sentiment classification results are obtained. We can observe that even with only 25% training data, our model still beats most baselines in Table 2. However, the perplexity increases slightly with the increased size of the training data. Since we do not have any topic coherence relevant terms in the model objective function, the coherence scores do not necessarily increase with the increasing size of the training data. But we can still observe a general upward trend of the topic coherence scores.

4.5 Sentiment Classification on the Full Dataset

In this section, we explore the performance of sentiment classification on the full dataset. Here, we follow the experimental setup in [32] and report the results in Table 6. For our proposed method, the number of topics is 300 and the vocabulary size for the topic model is 5,000. The results of referenced methods are similar to those reported on the sampled dataset shown in Table 2. Among the baselines, NSC beats other models by a large margin. Among our proposed approaches, we notice that using hard attention (MTL-HA) performs worse since the topic distribution is not well trained and simply using mean pooling introduces noises into the sentiment classifier. The performance of mutual learning based methods is similar to that on the sampled data. MTL-ML$_3$ and MTL-ML$_4$ beat MTL-ML$_1$ and MTL-ML$_2$ by a small margin.

When comparing our proposed approaches with NSC, we can see that MTL-ML$_4$ outperforms NSC on the Yelp 2013 and IMDB datasets while MTL-ML$_3$ gives slightly better results on the Yelp 2014 dataset. We observe that in the largest dataset, Yelp 2015, using mutual learning performs slightly worse than NSC by 0.7%. This is due to the training constraint imposed on topic modeling that only limited vocabulary can be used since the input to the neural topic model is the bag-of-words representation. In future work, we will explore other ways in formulating neural topic models such as based on Bayesian formulation of the skip-gram model in order to deal with large-scale training data.

5 Conclusion

In this paper, we have presented a new Multi-Task Learning framework with mutual learning for both sentiment classification and topic extraction. We evaluate our model on subsets of the Yelp and IMDB datasets and show that our model achieves superior performance compared with several strong baselines on sentiment classification. On the original full datasets, our proposed method still beats the baselines most of the time even though the topic modeling component can only be trained with a reduced vocabulary. The proposed framework also extracts more semantically coherent topics. In future work, we will explore extending this model for sentiment/topic dynamics analysis.

Acknowledgments

This work was supported by EU-H2020 (grant no. 794196), EPSRC (grant no. EP/T017112/1), Innovate UK (grant no. 103652), National Natural Science Foundation of China 61876053, 61632011, Shenzhen Foundational Research Funding JCYJ20180507183527919.

References

[1] R. M. Neal, "Probabilistic inference using markov chain monte carlo methods," 1993.
[2] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan, "An introduction to MCMC for machine learning," *Machine Learning*, vol. 50, no. 1-2, pp. 5–43, 2003.
[3] C. Lin and Y. He, "Joint sentiment/topic model for sentiment analysis," in *The ACM Conference on Information and Knowledge Management*, 2009, pp. 375–384.
[4] C. Lin, Y. He, R. Everson, and S. Ruger, "Weakly supervised joint sentiment-topic detection from text," *IEEE Transactions on Knowledge and Data engineering*, vol. 24, no. 6, pp. 1134–1145, 2012.
[5] C. Ma, M. Wang, and X. Chen, "Topic and sentiment unification maximum entropy model for online review analysis," in *Proceedings of the 24th International Conference on World Wide Web*. ACM, 2015, pp. 649–654.
[6] J. Li, M. Liao, W. Gao, Y. He, and K.-F. Wong, "Topic extraction from microblog posts using conversation structures," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2016, pp. 2114–2123.
[7] V. Rakesh, W. Ding, A. Ahuja, N. Rao, Y. Sun, and C. K. Reddy, "A sparse topic model for extracting aspect-specific summaries from online reviews," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2018, pp. 1573–1582.
[8] X. Li, J. Chi, C. Li, J. Ouyang, and B. Fu, "Integrating topic modeling with word embeddings by mixtures of vmfs," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 151–160.
[9] K. Batmanghelich, A. Saeedi, K. Narasimhan, and S. Gershman, "Nonparametric spherical topic modeling with word embeddings," *arXiv preprint arXiv:1604.00126*, 2016.
[10] R. Das, M. Zaheer, and C. Dyer, "Gaussian lda for topic models with word embeddings," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, 2015, pp. 795–804.
[11] P. Xie, Y. Deng, and E. P. Xing, "Diversifying restricted boltzmann machine for document modeling," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, 2015, pp. 1315–1324.
[12] R. He, "An unsupervised neural attention model for aspect extraction," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, 2017, pp. 388–397.
[13] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier, "Effective attention modeling for aspect-level sentiment classification," in *Proceedings of the 27th International Conference on Computational Linguistics, COLING*, 2018, pp. 1121–1131.
[14] R. Das, M. Zaheer, and C. Dyer, "Gaussian LDA for topic models with word embeddings," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, 2015, pp. 795–804.
[15] P. Xie, J. Zhu, and E. P. Xing, "Diversity-promoting bayesian learning of latent variable models," in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, 2016, pp. 59–68.
[16] M. Peng, Q. Xie, H. Wang, Y. Zhang, X. Zhang, J. Huang, and G. Tian, "Neural sparse topical coding," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, 2018, pp. 2332–2340.
[17] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2013. [Online]. Available: http://arxiv.org/abs/1312.6114
[18] Y. Miao, L. Yu, and P. Blunsom, "Neural variational inference for text processing," in *The International Conference on Machine Learning*, 2016, pp. 1727–1736.
[19] D. Card, C. Tan, and N. A. Smith, "A neural framework for generalized topic models," *arXiv preprint arXiv:1705.09296*, 2017.
[20] A. Srivastava and C. Sutton, "Autoencoding variational inference for topic models," in *International Conference on Learning Representations*, 2017.

Method	25% Training data			50% Training data			75% Training data		
	Accuracy	**PPL**	C_v	Accuracy	**PPL**	C_v	Accuracy	**PPL**	C_v
Data	Yelp								
MTL-ML$_1$	0.549	1056.3	0.466	0.571	1191.5	0.496	0.609	1269.4	0.517
MTL-ML$_2$	0.557	1039.0	0.524	0.591	1165.5	0.543	0.607	1262.1	0.500
MTL-ML$_3$	0.589	1091.5	0.512	0.612	1197.7	0.494	0.616	1264.6	0.554
MTL-ML$_4$	0.579	1092.7	0.540	0.614	1196.8	0.535	0.620	1263.9	0.595
Data	IMDB								
MTL-ML$_1$	0.324	758.5	0.444	0.347	859.5	0.511	0.358	897.6	0.536
MTL-ML$_2$	0.321	756.3	0.522	0.326	858.1	0.567	0.355	895.9	0.543
MTL-ML$_3$	0.367	760.5	0.575	0.386	860.7	0.598	0.396	897.8	0.643
MTL-ML$_4$	0.361	761.3	0.546	0.375	861.4	0.566	0.395	899.9	0.585

TABLE 5
The performance of sentiment classification and topic generation by our proposed method with different sizes of the training data.

Model	Yelp 2013	Yelp 2014	Yelp 2015	IMDB
LightGBM	56.8	57.5	58.2	31.6
CNN+	63.7	61.2	64.7	41.3
RNN	62.1	62.5	63.1	41.1
CRNN	63.7	65.5	66.0	42.5
NSC	67.3	69.5	**69.8**	49.0
MTL-RNN	67.2	69.2	69.7	48.3
MTL-HA	65.7	63.2	66.7	46.3
MTL-ML$_1$	67.1	68.9	68.8	48.2
MTL-ML$_2$	66.9	69.4	68.6	48.4
MTL-ML$_3$	67.4	**70.2**	69.6	49.3
MTL-ML$_4$	**68.6**	70.1	69.4	**49.7**

TABLE 6
Sentiment classification accuracy of our proposed against baselines on full dataset[15].

[21] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther, "Ladder variational autoencoders," in *The Annual Conference on Neural Information Processing Systems*, 2016, pp. 3738–3746.

[22] D. Bouchacourt, R. Tomioka, and S. Nowozin, "Multi-level variational autoencoder: Learning disentangled representations from grouped observations," in *AAAI Conference on Artificial Intelligence*, 2018.

[23] S. Chaidaroon and Y. Fang, "Variational deep semantic hashing for text documents," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, 2017, pp. 75–84.

[24] M. Huang, Y. Rao, Y. Liu, H. Xie, and F. L. Wang, "Siamese network-based supervised topic modeling," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 4652–4662.

[25] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," in *The 2002 Conference on Empirical Methods on Natural Language Processing*, 2002, pp. 79–86.

[26] S. Wang and C. D. Manning, "Baselines and bigrams: Simple, good sentiment and topic classification," in *Proceedings of the 50nd Annual Meeting of the Association for Computational Linguistics*, 2012, pp. 90–94.

[27] E. Cambria, S. Poria, A. Gelbukh, and M. Thelwall, "Sentiment analysis is a big suitcase," *IEEE Intelligent Systems*, vol. 32, no. 6, pp. 74–80, 2017.

[28] Y. Dang, Y. Zhang, and H. Chen, "A lexicon-enhanced method for sentiment classification: An experiment on online product reviews," *IEEE Intelligent Systems*, vol. 25, no. 4, pp. 46–53, 2010.

[29] Y. Kim, "Convolutional neural networks for sentence classification," in *The 2014 Conference on Empirical Methods on Natural Language Processing*. Association for Computational Linguistics, 2014, pp. 1746–1751.

[30] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[31] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *The 2015 Conference on Empirical Methods on Natural Language Processing.*, 2015, pp. 1422–1432.

[32] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy, "Hierarchical attention networks for document classification." in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489.

[33] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, vol. 1, 2018, pp. 2227–2237.

[34] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[35] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[36] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," *arXiv preprint arXiv:1509.00685*, 2015.

[37] B. Dhingra, H. Liu, Z. Yang, W. Cohen, and R. Salakhutdinov, "Gated-attention readers for text comprehension," in *Proceedings of the 55nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2017, pp. 1832–1846.

[38] S. Wang, S. Mazumder, B. Liu, M. Zhou, and Y. Chang, "Target-sensitive memory networks for aspect sentiment classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018, pp. 957–967.

[39] H. Chen, M. Sun, C. Tu, Y. Lin, and Z. Liu, "Neural sentiment classification with user and product attention," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2016, pp. 1650–1659.

[40] D. Tang, B. Qin, and T. Liu, "Aspect level sentiment classification with deep memory network," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2016, pp. 214–224.

[41] C. Fan, Q. Gao, J. Du, L. Gui, R. Xu, and K. Wong, "Convolution-based memory network for aspect-based sentiment analysis," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, 2018, pp. 1161–1164.

[42] P. Chen, Z. Sun, L. Bing, and W. Yang, "Recurrent attention network on memory for aspect sentiment analysis," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, 2017, pp. 452–461.

[43] Y. Zou, T. Gui, Q. Zhang, and X. Huang, "A lexicon-based supervised attention model for neural sentiment analysis," in *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, 2018, pp. 868–877.

[44] J. Li, H. Yang, and C. Zong, "Document-level multi-aspect sentiment classification by jointly modeling users, aspects, and overall ratings," in *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, 2018, pp. 925–936.

[45] P. Liu, X. Qiu, and X. Huang, "Adversarial multi-task learning for text classification," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[46] M. Isonuma, T. Fujino, J. Mori, Y. Matsuo, and I. Sakata, "Extractive summarization using multi-task learning with document classification," in *The 2017 Conference on Empirical Methods on Natural Language Processing*, 2017, pp. 2101–2110.

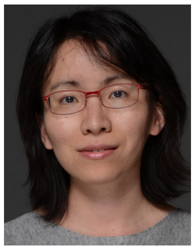
[47] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.
[48] A. Søgaard and Y. Goldberg, "Deep multi-task learning with low level tasks supervised at lower layers," in *Proceedings of the 54nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2016, pp. 231–235.
[49] H. M. Alonso and B. Plank, "When is multitask learning effective? semantic sequence prediction under varying data conditions," pp. 44–53, 2017.
[50] K. Singla, D. Can, and S. Narayanan, "A multi-task approach to learning multilingual representations," in *Proceedings of the 56nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 214–220.
[51] Y. Zhang and Q. Yang, "A survey on multi-task learning," *arXiv preprint arXiv:1707.08114*, 2017.
[52] P. Liu, X. Qiu, and X. Huang, "Recurrent neural network for text classification with multi-task learning," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, 2016, pp. 2873–2879.
[53] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
[54] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *The Annual Conference on Neural Information Processing Systems*, 2017, pp. 3146–3154.
[55] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity in phrase-level sentiment analysis," in *HLT/EMNLP 2005, Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 6-8 October 2005, Vancouver, British Columbia, Canada*, 2005, pp. 347–354.
[56] L. Gui, R. Xu, Y. He, Q. Lu, and Z. Wei, "Intersubjectivity and sentiment: From language to knowledge," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, 2016, pp. 2789–2795.
[57] M. Röder, A. Both, and A. Hinneburg, "Exploring the space of topic coherence measures," in *The 8th ACM International Conference on Web Search and Data Mining*, 2015, pp. 399–408.
[58] P. Mattei and J. Frellsen, "Leveraging the exact likelihood of deep latent variable models," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, 2018, pp. 3859–3870.

Lin Gui is a Marie Curie Research Fellow in the Department of Computer Science, University of Warwick, UK. He has published over 20 papers in top conferences and high-impact journals. His research interests include natural language processing, machine learning, text and data mining, sentiment analysis, and social media analysis. Lin obtained his PhD degree from Harbin Institute of Technology, China.

Jia Leng is a Research and Development Engineer in Meituan. Her research interests include natural language processing, sentiment analysis and topic modeling. She obtained her MEng degree from Harbin Institute of Technology.

Jiyun Zhou is now a postdoc in Department of Psychiatry and Behavioral Sciences, the Johns Hopkins University. His main research interests are bioinformatics, natural language processing and machine learning. He obtained his PhD degree from Hong Kong Polytechnic University.

Ruifeng Xu is a Professor in the School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), China. His research interests are natural language processing, emotion computing and text mining. He obtained his BEng degree from Harbin Institute of Technology, China, and MPhil and PhD degrees from Hong Kong Polytechnic University.



Yulan He is a Professor in the Department of Computer Science, University of Warwick, UK. She has published more than 160 papers with most appeared in high-impact journals and top conferences. Her research interests include natural language processing, statistical modeling, text and data mining. Yulan received her PhD degree in spoken language understanding from the University of Cambridge, UK.