Learning to Disentangle Interleaved Conversational Threads with a Siamese Hierarchical Network and Similarity Ranking

Jyun-Yu Jiang[†], **Francine Chen**[‡], **Yan-Ying Chen**[‡] **and Wei Wang**[†]
[†]University of California, Los Angeles, CA, USA
[‡]FX Palo Alto Laboratory, Palo Alto, CA, USA
jyunyu@cs.ucla.edu, {chen,yanying}@fxpal.com, weiwang@cs.ucla.edu

Abstract

An enormous amount of conversation occurs online every day, such as on chat platforms where multiple conversations may take place concurrently. Interleaved conversations lead to difficulties in not only following discussions but also retrieving relevant information from simultaneous messages. Conversation disentanglement aims to separate intermingled messages into detached conversations.

In this paper, we propose to leverage representation learning for conversation disentanglement. A Siamese hierarchical convolutional neural network (SHCNN), which integrates local and more global representations of a message, is first presented to estimate the conversation-level similarity between closely posted messages. With the estimated similarity scores, our algorithm for conversation identification by similarity ranking (CISIR) then derives conversations based on high-confidence message pairs and pairwise redundancy. Experiments were conducted with four publicly available datasets of conversations from Reddit and IRC channels. The experimental results show that our approach significantly outperforms comparative baselines in both pairwise similarity estimation and conversation disentanglement.

1 Introduction

With the growth of ubiquitous internet and mobile devices, people now commonly communicate in the virtual world. Among the various methods of communication, text-based conversational media, such as internet relay chat (IRC) (Werry, 1996) and Facebook Messenger[1], has been and remains one of the most popular choices. In addition, many enterprises have started to use conversational chat platforms such as Slack[2] to enhance team collaboration. However, multiple conversations may

Thread	Message
⋮	⋮
T31	*Malcolm: If running as root, I need to set up a global config rather than ~/.fetchmailrc ?*
T38	***Elma:*** *i'm sure i missed something but fonts rendering in my gimp works isn't at its best*
T39	*Sena: is there anyway to see what the CPU temperature is?*
T38	***Elma:*** *is it because of gimp or i missed some tuning or such?*
T31	*Rache: Specify a non-default name run control file.*
T41	*Denny: so how does one enforce a permission set and ownership set on a folder and all its children?*
T31	*Malcolm: in the man page it doesn't mention any global fetchmailrc file... that is what was confusing me...*
T42	*Shenna: hi, are sata drives accessed as sda or hda?*
T41	***Elma:*** *-R for recursive...*
T42	***Elma:*** *sda*
⋮	⋮

Figure 1: A segment of real-world conversations involving six users and five (annotated) threads from the IRC dataset.

occur simultaneously when conversations involve three or more participants. Aoki et al. (2006) found an average of 1.79 conversations among eight participants at a time. Moreover, some platforms like chatrooms in Twitch may have more concurrent conversations (Hamilton et al., 2014). Interleaved conversations can lead to difficulties in both grasping discussions and identifying messages related to a search result. For example, Figure 1 shows a segment of conversations from the real-world IRC dataset as an example. Five interleaved threads are involved in only ten messages. Messages in the same thread may not have identical keywords. Moreover, a user (i.e., *Elma*) can participate in multiple threads. Hence, a robust mechanism to disentangle interleaved conversations can improve a user's satisfaction with a chat system.

One solution for conversation disentanglement is to model the task as a topic detection and tracking (TDT) (Allan, 2002) task by deciding whether each incoming message starts a new topic or belongs to an existing conversation. Messages in the same conversation may have higher similarity

[1]Facebook Messenger: https://www.messenger.com/
[2]Slack: https://slack.com/

scores (Shen et al., 2006; Mayfield et al., 2012) or similar context messages (Wang and Oard, 2009). However, similarity thresholds for determining new topics vary depending on context. Embedding of earlier messages, resulting in duplication of parts of messages, can alter the similarity score. More specifically, the similarity scores obtained in previous work cannot well represent conversation-level relationships between messages.

Several studies have examined the use of statistical (Du et al., 2017) and linguistic features (Elsner and Charniak, 2008, 2010, 2011; Mayfield et al., 2012) for predicting user annotations of paired message similarity. These studies employed bag-of-words representations which do not capture term similarity and cannot distinguish word importance and relationships between words in a message. Thus, better representations of messages and their relationships are needed.

Recent studies have demonstrated the effectiveness of deep learning methods in representation learning (Bengio et al., 2013), aiming to infer low-dimensional distributed representations for sparse data such as text (Hinton and Salakhutdinov, 2006). These representations can be derived not only for words (Mikolov et al., 2013) but also sentences and documents (Le and Mikolov, 2014). In particular, convolutional neural networks (CNNs) have been shown to efficiently and effectively preserve important semantic and syntactic information from embedded text sequences (Blunsom et al., 2014). It has been demonstrated that CNNs produce state-of-the-art results in many NLP tasks such as text classification (Kim, 2014; Lai et al., 2015; Zhang et al., 2015) and sentiment analysis (Tang et al., 2014; Poria et al., 2015). Existing approaches, however, do not take advantage of deep learning techniques to model relationships between messages for disentangling conversations. (Mehri and Carenini, 2017) defined many statistical features for use with a random forest for in-thread classification and used a recurrent neural network (RNN) only to model adjacent messages with an external dataset as a feature.

In this paper, we aim to leverage deep learning for conversation disentanglement. Our proposed approach consists of two stages: (1) message pair similarity estimation and (2) conversation identification. In the first stage, we propose the Siamese hierarchical convolutional neural network (SHCNN) to estimate conversation-level similarity between pairs of closely posted messages. SHCNN is framed as a Siamese architecture (Mueller and Thyagarajan, 2016) concatenating the outputs of two hierarchical convolutional neural networks and additional features. Compared to other conventional CNN-based Siamese networks (Severyn and Moschitti, 2015; Yin et al., 2016), SHCNN models not only local information in adjacent words but also more global semantic information in a message. In the second stage, the algorithm of conversation identification by similarity ranking (CISIR) ranks messages within a time window paired with each message and constructs a message graph involving high-rank connections with strong confidence. Although only high-confidence relations are represented in the constructed graph, the redundancy of pairwise relationships can capture the connectivity of messages within a conversation.

In summary, the main contributions of this paper are threefold: **(1) Deep similarity estimation for conversation disentanglement:** To the best of our knowledge, this is the first study applying deep learning to estimate similarities between messages for disentangling conversations. SHCNN simultaneously captures and compares local and global characteristics of two messages to estimate their similarity. Message representations are also optimized towards the task of conversation disentanglement. **(2) Efficient and effective method:** The selection of message pairs posted closely in time and the proposed CISIR algorithm significantly reduces the computational time from $O\left(|M|^2\right)$ to $O\left(k|M|\right)$, where $|M|$ is the number of messages, and k is the maximum number of messages posted within a fixed-length time window. When many messages are posted over a long period, the computational time of our approach could be near-linear. **(3) Empirical improvements over previous work:** Extensive experiments have been conducted on four publicly available datasets, including three synthetic conversation datasets and one real conversation dataset from Reddit[3] and IRC conversations. Our approach outperforms all comparative baselines for both similarity estimation and conversation disentanglement.

2 Related Work

Methods for conversation disentanglement can be simply categorized into unsupervised and supervised approaches. Unsupervised approaches (Wang and Oard, 2009) estimate the relationship between messages through unsupervised similarity functions such cosine similarity, and assign messages to conversations based on a predefined

[3]Reddit: `https://www.reddit.com/`

threshold. In contrast, supervised methods exploit a set of user annotations (Elsner and Charniak, 2008; Mayfield et al., 2012; Shen et al., 2006; Du et al., 2017; Mehri and Carenini, 2017) to adapt to different datasets. Our approach can be classified as a supervised approach because a small set of user annotations is used to train the SHCNN.

In addition to conversations, some studies predict the partial structure of threaded data, especially for online forums (Aumayr et al., 2011; Wang et al., 2011b,a). These studies merely classify parent-child relationships in disentangled, independent threads. Moreover, they focus only on comments to the same post. Indeed, conversation disentanglement is a more difficult task.

Estimating the similarity of text pairs is an essential part in our approach. Many studies also focus on similar tasks aside from conversation disentanglement, such as entailment prediction (Mueller and Thyagarajan, 2016; Wang and Jiang, 2017) and question-answering (Severyn and Moschitti, 2015; Amiri et al., 2016; Yin et al., 2016). However, most of their models are complicated and require a larger amount of labeled training data; limited conversational data can lead to unsatisfactory performance as shown in Section 4.

3 Conversation Disentanglement

In this section, we formally define the objective of this work and notations used. A two-stage approach is then proposed to address the problem.

3.1 Problem Statement

Given a set of speakers $\boldsymbol{S}$, a message m is defined as a tuple $m = (\boldsymbol{w}, s, t)$, where $\boldsymbol{w} = \langle w_1, w_2, \cdots, w_n \rangle$ is a word sequence posted by the speaker $s \in \boldsymbol{S}$ at time t in seconds. Each message m is associated with a conversation $z(m)$. Messages in different conversations can be posted concurrently, i.e., conversations can be interleaved.

Following the settings of previous work (Elsner and Charniak, 2008, 2010, 2011; Mayfield et al., 2012), a set of pairwise annotations $\boldsymbol{A} = \{(m_i, m_j, y)\}$, where $y \in \{0, 1\}$, is given for training the model. More specifically, a Boolean value y indicates whether two messages m_i and m_j are in the same conversation, i.e., $z(m_i)$ and $z(m_j)$ are identical.

Given a set of messages $\boldsymbol{M}$ and the pairwise annotations $\boldsymbol{A}$ as training data, the goal is to learn a model that can identify whether messages are posted in the same conversation $z(m)$. Note that the number of conversations $|\boldsymbol{Z} = \{z(m) \mid \forall m \in \boldsymbol{M}\}|$ is always unknown to the system.

3.2 Framework Overview

Figure 2 illustrates our two-stage framework. The first stage aims to estimate pairwise similarity among messages. Message pair selection is applied to focus on the similarity between messages that are posted closely in time and thus more likely to be in the same conversation. The Siamese hierarchical CNN (SHCNN) is proposed for learning message representations and estimating pairwise similarity scores. The overlapping hierarchical structure of SHCNN models a message at multiple semantic levels and obtains representations that are more comprehensive.

In the second stage, our conversation identification by similarity ranking (CISIR) algorithm exploits the redundancy and connectivity of pairwise relationships to identify conversations as connected components in a message graph.

3.3 Message Pair Selection

Most of the previous work on conversation disentanglement focused on pairwise relationships between messages (Mayfield et al., 2012). Especially for single-pass clustering approaches, all pairs of messages need to be enumerated during similarity computation (Wang and Oard, 2009). However, if messages have been collected for a long time, the number of message pairs could be too mammoth to be processed in an acceptable amount of time. More precisely, it leads to at least $O(n^2)$ computational time, where n is the number of messages. As shown in Figure 3, the percentage of messages in the same conversation as a given message becomes significantly lower with a longer elapsed time between consecutive messages. In light of this observation, an assumption is made as follows:

Assumption 1 *The elapsed time between two consecutive messages posted in the same conversation is not greater than T hours, where T is a small number.*

More specifically, in our dataset every message m_i is posted within T hours earlier or later than any other message m_j in the same conversation, i.e., $\frac{|t_i - t_j|}{3600} < T$ for all pairs (m_i, m_j), where t is in seconds. For example, in the IRC dataset the average elapsed time between consecutive messages in a conversation is only 7 minutes. If a conversation is ongoing, there may not be an extended silence before a new message; conversely, an extended silence could be treated as the start of a new

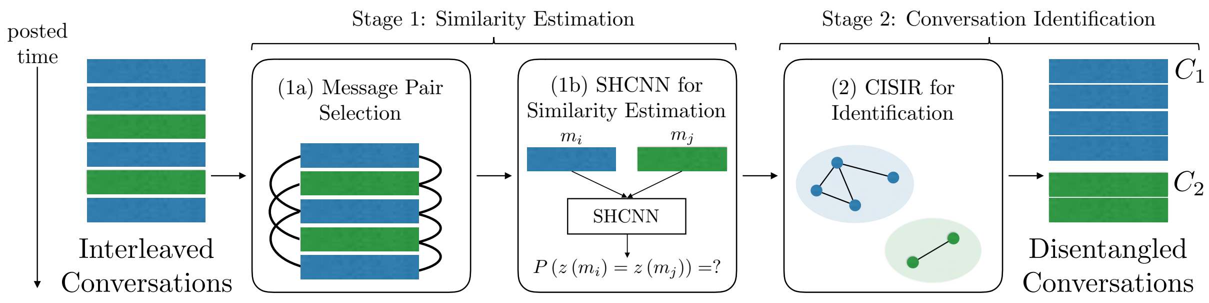
posted time
Interleaved Conversations
Stage 1: Similarity Estimation
(1a) Message Pair Selection
(1b) SHCNN for Similarity Estimation
m_i
m_j
SHCNN
$P(z(m_i) = z(m_j)) = ?$
Stage 2: Conversation Identification
(2) CISIR for Identification
C_1
C_2
Disentangled Conversations

Figure 2: Illustration of our proposed two stage method. In the first stage, (1a) message pairs are selected for (1b) estimating pairwise similarity with a Siamese hierarchical CNN (SHCNN). In the second stage, (2) the algorithm of conversation identification by similarity ranking (CISIR) constructs a graph with strong relationships among messages and finds conversations as connected components.

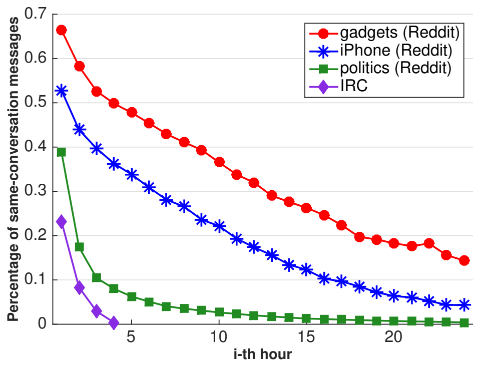

Figure 3: The percentage of messages in the same conversation as a given message with elapsed time between messages no greater than *i* hours for four experimental datasets.

conversation. With this assumption, the number of pairs can be reduced to $O(kn)$, where k is the maximum number of messages posted in a T-hour time window. By default T is set to 1 hour in our experiments.

In addition, it is worth mentioning that it may be possible to include conversational structure, such as replied-to relations, into the model. For example, after using CISIR to identify conversational threads, structure inference may be performed using methods such as described in (Aumayr et al., 2011) or (Wang et al., 2011b) and the structure used to refine the threads. In this study, we focus on only conversation disentanglement.

3.4 Similarity Estimation with the Siamese Hierarchical CNN (SHCNN)

Given a set of message pairs, we propose the Siamese hierarchical CNN (SHCNN) to estimate the similarity between a pair of messages.

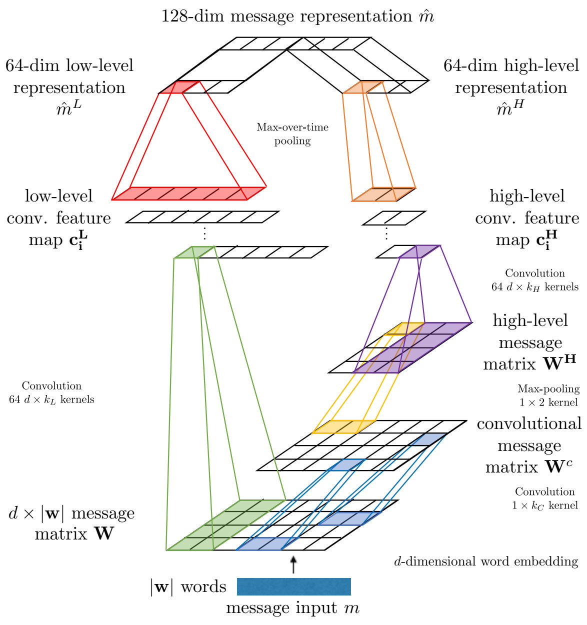
128-dim message representation $\hat{m}$
64-dim low-level representation $\hat{m}^L$
64-dim high-level representation $\hat{m}^H$
Max-over-time pooling
low-level conv. feature map $\mathbf{c}_i^L$
high-level conv. feature map $\mathbf{c}_i^H$
Convolution 64 $d \times k_H$ kernels
high-level message matrix $\mathbf{W}^H$
Convolution 64 $d \times k_L$ kernels
Max-pooling 1×2 kernel
convolutional message matrix $\mathbf{W}^c$
Convolution $1 \times k_C$ kernel
$d \times |\mathbf{w}|$ message matrix $\mathbf{W}$
d-dimensional word embedding
$|\mathbf{w}|$ words
message input m

Figure 4: Illustration of hierarchical CNN (HCNN) for message representation. The labels with a larger font size indicate the corresponding tensors, and the labels with a smaller font size explain the operations between tensors.

3.4.1 Hierarchical CNN for Message Representation

The effectiveness of CNNs for representing text has already been addressed in previous studies. However, single-layer CNNs (Kim, 2014; Severyn and Moschitti, 2015) may not represent high-level semantics while low-level information could be diluted with multiple-layer CNNs (Yin et al., 2016). The hierarchical CNN (HCNN) is designed to simultaneously capture low- and high-level message meanings as shown in Figure 4.

A message m_i is first represented by a $d \times |\boldsymbol{w}|$ message matrix $\boldsymbol{W} \in \boldsymbol{R}^{d \times |\boldsymbol{w}|}$, where d is the dimension of a word embedding, and $|\boldsymbol{w}|$ is the number of words in a message. For low-level information, we exploit single-layer CNNs (Kim, 2014; Severyn and Moschitti, 2015) with a set of $d \times k_L$ kernels, where L denotes "Low", to extract n-gram semantics of k_L contiguous words. In this paper, 64 $d \times k_L$ kernels, where $k_L = 5$, are applied to obtain 64 low-level features $\hat{m}^L$. Note that the kernel row dimension is identical to the word embedding dimension to jointly consider the full embedding vector. As a consequence, convolution with each kernel produces a vector $\boldsymbol{c_i^L}$, which is then aggregated by max-over-time pooling (Collobert et al., 2011; Kim, 2014).

To acquire high-level semantics across a message, HCNN uses another multiple-layer CNN for feature extraction. A $1 \times k_C$ kernel is applied to $\boldsymbol{W}$, thereby generating a convolutional message matrix $\boldsymbol{W^C}$. Features covering broader contents are computed by applying a 1×2 kernel to a max-pooling layer with a stride of 2, producing a high-level message matrix $\boldsymbol{W^H}$. The row sizes of the two kernels are set to 1 to capture relations within each embedding dimension, and convolution is performed on $\boldsymbol{W^H}$ with 64 $d \times k_H$ kernels to capture relations across embedding dimensions. The generated convolutional feature maps $\boldsymbol{c_i^H}$ are subject to max-over-time pooling, resulting in 64 features $\hat{m}^H$. Finally, a message representation $\hat{m}$ is constructed by concatenating $\hat{m}^L$ and $\hat{m}^H$, i.e., creating a 128-dimensional feature vector, for characterizing both low- and high-level semantics of a message m. In this paper, both k_C and k_H are set to 5 while computing high-level representations.

3.4.2 Siamese Hierarchical CNN (SHCNN)

A Siamese structure with two identical sub-networks is useful to exploit the affinity between representations of two instances in the same hidden space (Severyn and Moschitti, 2015; Yin et al., 2016; Wang and Jiang, 2017). For similarity estimation, we propose the Siamese hierarchical CNN (SHCNN) using a Siamese structure that blends the outputs from two HCNNs as well as some context features.

Figure 5 shows the structure of the SHCNN for estimating the similarity between two messages m_i and m_j where the message representations $\hat{m}_i$ and $\hat{m}_j$ are generated by two sub-networks HCNNs (See Figure 4). There are many ways to deal with two sub-networks, such as using a similarity matrix (Severyn and Moschitti, 2015) or an attention matrix (Yin et al., 2016). However, both methods lead to an enormous number of parameters for long messages. We propose to independently compute the element-wise absolute differences (Mueller and Thyagarajan, 2016) between a pair of message representations $\hat{m}_i$ and $\hat{m}_j$, each from a sub-network. More formally, the absolute difference d is a vector where the k-th element is computed as $|\hat{m}_i(k) - \hat{m}_j(k)|$. This approach provides not only fewer parameters but also the flexibility to observe interactions among different dimensions in representations. Our experiments also show it outperforms the other two approaches in similarity estimation (See Section 4).

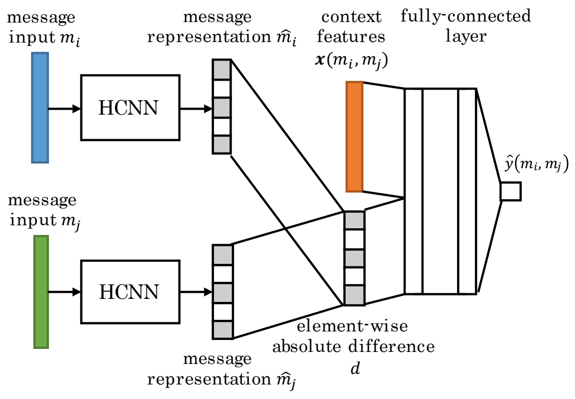
message input m_i
message representation $\hat{m}_i$
context features $\boldsymbol{x}(m_i, m_j)$
fully-connected layer
HCNN
$\hat{y}(m_i, m_j)$
message input m_j
HCNN
element-wise absolute difference d
message representation $\hat{m}_j$

Figure 5: The siamese hierarchical CNN (SHCNN) for similarity estimation. Note that the model structure of an HCNN is shown in Figure 4.

In addition to message contents, contexts such as temporal and user information were also usually considered in previous studies about conversation disentanglement (Wang and Oard, 2009; Elsner and Charniak, 2010, 2011). In this paper, we focus on the performance of message content representations and only incorporate four context features: speaker identicality, absolute time difference and the number of duplicated words with and without weighting by inverse document frequency (Christopher et al., 2008). SHCNN concatenates the context features $\boldsymbol{x}(m_i, m_j)$ with the absolute difference d as the input of a fully-connected layer of the same size.

The final output of SHCNN $\hat{y}(m_i, m_j)$ is normalized by a logistic sigmoid function (Han and Moraga, 1995), representing the probability $P(z(m_i) = z(m_j))$.

3.4.3 Activation Functions

All convolutional layers and the fully-connected layer require activation functions, and the choice affects the performance (Maas et al., 2013). Popular functions include rectified linear units (ReLUs) (LeCun et al., 2015), hyperbolic tangent units ($\tanh$) and exponential linear units (ELUs) (Clevert et al., 2016). In this study, we conducted informal comparison experiments and ELU was finally chosen for all functions because it performed the best.

3.4.4 Optimization and Implementation Details

Given a set of annotated message pairs $\boldsymbol{A} = \{(m_i, m_j, y)\}$, where y is a Boolean value indicating whether two messages are in the same conversation, SHCNN is optimized with binomial cross entropy (Goodfellow et al., 2016). More formally, the objective function is as follows:

$$\sum_{(m_i,m_j,y)\in A}[y \cdot \log(\hat{y}+\epsilon) + (1-y)\cdot\log(1-\hat{y}+\epsilon)] + \lambda||\boldsymbol{\theta}||^2$$

where $\hat{y}$ simplifies $\hat{y}(m_i, m_j)$, and ϵ is a small number, i.e., 10^{-9} in our experiments, preventing underflow errors. The term λ serves as the weight for L2-regularization for the set of parameters $\boldsymbol{\theta}$.

In our experiments, SHCNN is implemented by TensorFlow (Abadi et al., 2016) and trained by the Adam optimizer (Kingma and Ba, 2015) with an initial learning rate of 10^{-3}. The dropout technique (Srivastava et al., 2014) is utilized in the fully-connected layer with a dropout probability of 0.1. Word embeddings are initialized using the publicly available fastText 300-dimensional pretrained embeddings from Facebook (Bojanowski et al., 2016). The batch size is set to 512, and the maximum number of training epochs is 1,000. The final model is determined by evaluating the mean average precision (MAP) on a validation dataset every 100 iterations.

3.5 Conversation Identification by SImilarity Ranking (CISIR)

In the second stage of conversation disentanglement, i.e., part (2) in Figure 2, we aim to separate conversations based on the identified message pairs and their estimated similarity.

3.5.1 Graph-based Methods and Conversation Connectivity

It is intuitive to apply graph-based methods if pairwise relationships of messages are exploited (Elsner and Charniak, 2008). Furthermore, methods based on single-pass clustering (Wang and Oard, 2009) can be also be treated as graph-based methods. However, graph-based methods have a risky drawback: A single false positive connection between two messages can be propagated to several messages from different conversations. As shown in Figure 3, a certain percentage of message pairs are in different conversations, which can lead to numerous false positive connections.

Algorithm 1: The algorithm of conversation disentanglement by similarity ranking (CISIR).

```
1  CISIR ($\boldsymbol{M}, \boldsymbol{D}, r, h$);
   Input  : Message set $\boldsymbol{M}$, the set of selected
            message pairs $\boldsymbol{D}$, the threshold of
            similarity ranks $r$ and the threshold
            of similarity scores $h$.
   Output: A set of conversations $\boldsymbol{C}$
2  Let $G = (\boldsymbol{M}, \emptyset)$ be an undirected message
     graph
3  for $m \in M$ do
4      $\boldsymbol{D}_m =$
         $\{(m_i, m_j, \hat{y}) \mid m_i = m \vee m_j = m\}$
5      Rank entries in $\boldsymbol{D}_m$ by $\hat{y}$ in a descending
         order
6      for $k = 1$ to $\min(r, |\boldsymbol{D}_m|)$ do
7          Let $(m_i, m_j, \hat{y})$ be the $k$-th entry in
             ranked $\boldsymbol{D}_m$
8          if $\hat{y} < h$ then
9              break
10         Add an edge $(m_i, m_j)$ into $G$
11 $\boldsymbol{C}$ = ConnectedComponents($G$)
12 return $\boldsymbol{C}$
```

False alarms may be reduced by raising the threshold that determines whether two messages are connected (Wang and Oard, 2009). However, a high threshold can make disentangled conversations fragmented and the best threshold for each pair could vary.

3.5.2 The CISIR Algorithm

Instead of setting a high threshold, we propose the algorithm of Conversation Identification by SImilarity Ranking (CISIR). CISIR focuses on the top messages ranked by similarity scores. Based on Assumption 1, for each message, there exists at least one or more other messages in the same conversation posted closely in time. With this redundancy, a few pairs with stronger confidence, i.e., the top-ranked pairs, can be enough to extend a correct connectivity to earlier or later messages, while the low-ranked pairs can be ignored to reduce the risk of error propagation.

Given a set of selected message pairs with estimated similarity scores $\boldsymbol{D} = \{(m_i, m_j, \hat{y})\}$, Algorithm 1 shows the procedure of CISIR with two parameters r and h, where r is a high threshold of similarity ranks and h is a lower threshold of similarity scores. Note that CISIR filters out pairs with low scores because a message can have more than r same-conversation pairs posted in its T-hour time window. For each message, CISIR ranks all of its associated pairs by the estimated similarity and only retrieves the top-r pairs whose similarity scores are greater than h. These retrieved high-confidence pairs are treated as the edges in a message graph G. Finally, CISIR divides G into connected components, and the messages in each connected component are treated as a conversation. In this paper, we use grid search to set r and h as 5 and 0.5, respectively.

3.5.3 Improvement of Time Complexity

The efficiency of Algorithm 1 can be further improved. The top-r qualified pairs for each message can be pre-processed by a scan of $\boldsymbol{D}$ with $|M|$ min-heaps which always contain at most $r+1$ elements. When r is a small constant number, it only takes $O(|D|) = O(k \cdot |M|)$ for pre-processing, where k is the maximum number of messages posted in a T-hour time window. With pre-processed top pairs, CISIR can do graph construction and find connected components in $O(k|M|)$, which compares favorably to conventional methods in $O(|M|^2)$.

4 Experiments

In this section, we conduct extensive experiments on four publicly available datasets to evaluate SHCNN and CISIR in two stages.

4.1 Datasets and Experimental Settings

4.1.1 Datasets

Three datasets from Reddit and one dataset of IRC are used as the experimental datasets.

- **Reddit Datasets**[4] The Reddit dataset is comprised of all posts and corresponding comments in all sub-reddits (i.e., forums in Reddit.com) from June 2016 to May 2017. Comments under a post can be treated as messages in one conversational thread. Here we manually merge all comments in a sub-reddit to construct a synthetic dataset of interleaved conversations. Note that although it is called a "synthetic dataset," all messages are written by real users. Three subreddits with different popularity levels as shown in Table 1 are selected to build three datasets: gadgets, iPhone and politics.

[4]The organized Reddit dataset is publicly available in `https://files.pushshift.io/reddit/`.

Dataset	Reddit			IRC
	gadgets	iPhone	politics	
Conversations	287	617	3,671	39
Messages	8,518	12,433	105,663	497
Speakers	5,185	5,231	25,289	71
Train/Valid Pairs	3,445	5,556	244,492	5,995
Test Pairs	27,565	44,450	1,955,943	47,966

Table 1: Statistics of four datasets after pre-processing.

- **IRC Dataset.** An annotated IRC dataset used in (Elsner and Charniak, 2008) is also included in our experiments. The IRC dataset consists of about 6 hours of messages in interleaved conversations. Even though the IRC dataset is significantly smaller and shorter than the Reddit datasets, it consists of natural, interleaved conversations with ground truth annotations, including thread id.

4.1.2 Experimental Settings

Humans may not participate in a large number of simultaneous conversations. e.g., an average of 1.79 for eight people (Aoki et al., 2006), but there could be hundreds of concurrent posts in a subreddit. Hence, we adjusted the datasets to be more similar to real conversations. Specifically we removed some conversations so that every dataset has at most ten conversations at any point in time. Short messages with less than five words are also removed because even for humans they are frequently ambiguous. Too short conversations with less than ten messages are also discarded as outliers (Ren et al., 2011). Training and validation data are randomly chosen from only 10% of the selected message pairs, respectively, because in real situations obtaining labels could be very costly. The remaining 80% of pairs are regarded as testing data. As a result, Table 1 shows the statistics of the four datasets after pre-processing.

4.2 Pairwise Similarity Estimation

Message pair similarity estimation is treated as a ranking task and evaluated with three ranking evaluation metrics: precision at 1 (P@1), mean average precision (MAP) and mean reciprocal rank (MRR) (Christopher et al., 2008). We compare the performance with six baseline methods, including the difference of posted time (*TimeDiff*), sameness of speakers (*Speaker*), cosine similarity of text (*Text-Sim*), the approach proposed by Elsner and Charniak (2008) (Elsner), *DeepQA* (Severyn and Moschitti, 2015) and *ABCNN* (Yin et al., 2016). Note that DeepQA and ABCNN are neural network-based models for question-answering. The approach of Mehri and Carenini

Dataset	Reddit Datasets									IRC Dataset		
	gadgets			iPhone			politics					
Metric	P@1	MRR	MAP	P@1	MRR	MAP	P@1	MRR	MAP	P@1	MRR	MAP
TimeDiff	0.6916	0.8237	0.8170	0.6085	0.7651	0.7495	0.4412	0.6362	0.5644	0.3262	0.5180	0.4384
Speaker	0.5643	0.7046	0.7425	0.5364	0.6595	0.6590	0.4021	0.4620	0.3914	0.4356	0.6263	0.6891
Text-Sim	0.7913	0.8746	0.8440	0.7347	0.8318	0.7872	0.5245	0.6672	0.5326	0.3712	0.5269	0.3108
Elsner	0.7758	0.8651	0.8321	0.6809	0.7935	0.7471	0.4643	0.6132	0.4884	0.1094	0.1886	0.2063
DeepQA	0.8011	0.8755	0.8511	0.7156	0.8112	0.7766	0.5593	0.6759	0.5685	0.7811	0.8182	0.8050
ABCNN	0.8374	0.8511	0.8502	0.8112	0.8520	0.8118	0.7419	0.6221	0.6644	0.7008	0.4142	0.5858
SHCNN	**0.8834**	**0.9281**	**0.9005**	**0.8375**	**0.8944**	**0.8497**	**0.7696**	**0.8392**	**0.6967**	0.9785	**0.9838**	**0.9819**
SHCNN (L)	0.8470	0.9080	0.8702	0.8066	0.8792	0.8275	0.7225	0.8070	0.6438	**0.9807**	0.9834	0.9750
SHCNN (H)	0.8490	0.9105	0.8704	0.8158	0.8851	0.8313	0.7228	0.8110	0.6283	0.9635	0.9728	0.8632

Table 2: Performance of pairwise similarity estimation in four datasets. Our approach is denoted as SHCNN.The performance with only low-level or high-level representations are denoted as SHCNN (L) and SHCNN (H). All improvements of SHCNN against the best baseline are significant at the 1% level of significance in a paired t-test.

$z(m_i)$	Message
T16	*"**Arlie:** Wow, maybe we just missed it when we were driving around"*
T18	*"**Arlie:** i've been very close to that situation myself"*

Figure 6: An example message pair in two different conversations from IRC shows how SHCNN discriminates between messages on different topics. The leftmost column is the conversation IDs of the corresponding messages. SHCNN predicts 0.67% of being in the same conversation for this pair while DeepQA with single-layer CNNs predicts 69.81%.

$z(m_i)$	Message
T16	*"Very well, I seem to be trying to show Arlie how its done and am **coding a webserver**."*
T16	*"Arlie: Good enough doesnt cut it! Is **the 'faster' method** a big change in design? Could I **implement** later without wanting to kill myself?"*

Figure 7: An example message pair in a conversation from IRC shows how SHCNN captures similarity in local information. The leftmost column is the conversation IDs of the corresponding messages. SHCNN predicts 70.41% for this pair while ABCNN with multiple-layer CNNs predicts 36.50%.

(2017) was not compared in our experiments because the RNN requires additional message sequences; moreover, its performance was only mildly better than Elsner, which performed poorly on IRC in Table 2.

Table 2 shows the performance of similarity estimation. Among all methods, neural network approaches (Severyn and Moschitti, 2015; Yin et al., 2016) perform better than other methods in most cases, indicating that message content representation has considerable impact on estimating pairwise similarity. SHCNN outperforms most of the baselines even if only low-level (L) or high-level (H) representations are exploited. When SHCNN captures both low- and high-level semantics, it significantly outperforms all baselines across the four datasets. For example, ABCNN can outperform SHCNN using only either low- or high-level representations in the politics dataset; however, SHCNN turns the tables after using both representations. An interesting observation is that ABCNN is the best baseline in every dataset except for IRC; this may be because the IRC data is too small to train complicated attention structures. On the contrary, our SHCNN can precisely capture semantics even with few parameters and limited data.

To shed deeper insights of how SHCNN surpasses other methods, we exhibit the prediction results of the IRC data and demonstrate the capability of SHCNN to simultaneously preserve local and more global information. Figure 6 presents an example to show how SHCNN is better than other methods in capturing more high-level topical information. Even though the main sentences of two messages are clearly on different topics, the baseline method DeepQA (Severyn and Moschitti, 2015) still predicts a high similarity. This could be attributed to the context of author mention (Wang and Oard, 2009) and a bias on the local information, i.e., the exact same term "Arlie", in the Siamese network used in DeepQA. On the contrary, SHCNN can capture more global information that differentiates the topics and correctly predicts a very low score. Figure 7 illustrates another example of how SHCNN outperforms other methods in preserving the similarity of local information. Both of the messages in the example have some segments related to software engineering. A baseline method ABCNN (Yin et al., 2016) with multiple-layer CNNs, however, still predicts a low score. This might be because both sentences are long so that the local information is diluted after processing by multiple CNN layers. Differently, SHCNN is able to seize local information, correctly predicting a high score.

Dataset	Reddit Datasets									IRC Dataset		
	gadgets			iPhone			politics					
Metric	NMI	ARI	F1	NMI	ARI	F1	NMI	ARI	F1	NMI	ARI	F1
Doc2Vec	0.1757	0.0008	0.0589	0.2318	0.0002	0.0718	0.2672	0.0001	0.0506	0.2046	0.0048	0.1711
Block-10	0.7745	0.1840	0.3411	0.8203	0.2349	0.4251	0.8338	0.1724	0.3451	0.4821	0.0819	0.2087
Speaker	0.7647	0.0440	0.2094	0.7861	0.1001	0.3339	0.7480	0.0637	0.2207	0.7394	0.4572	0.6310
CBME	0.6913	0.0212	0.1465	0.7280	0.0339	0.1966	0.7883	0.0165	0.1382	0.2818	0.0324	0.1970
GTM	0.7942	0.1787	0.2986	0.8198	0.0536	0.2566	0.8496	0.3076	0.4292	0.0226	0.0001	0.2064
CISIR	**0.8254**	**0.4287**	**0.4939**	**0.8552**	**0.4236**	**0.5187**	**0.8825**	**0.3561**	**0.4950**	**0.9330**	**0.9543**	**0.8798**
Oracle	0.8608	0.4852	0.5560	0.9003	0.5448	0.6358	0.9651	0.8286	0.8863	0.9838	0.9850	0.9819

Table 3: Performance of conversation disentanglement in four datasets. Our approach is denoted as CISIR. "Oracle" indicates the optimal performance if CISIR correctly retrieves all message pairs in identical conversations. All improvements of CISIR against the best baseline are significant at the 1% level of significance in a paired t-test.

4.3 Conversation Identification

For conversation identification, three clustering metrics are adopted for evaluation: normalized mutual information (NMI), adjusted rand index (ARI) and F_1 score (F1). Six methods are implemented as the baselines for conversation disentanglement, including *Doc2Vec* (Le and Mikolov, 2014), blocks of 10 messages (*Block-10*), messages of respective speakers (*Speaker*) (Elsner and Charniak, 2011), context-based message expansion (*CBME*) (Wang and Oard, 2009) and a graph-theoretical model with chat- and content-specific features (Elsner and Charniak, 2008) (*GTM*). The embedding-based clustering method, i.e., *Doc2Vec*, applies affinity propagation (Frey and Dueck, 2007) to cluster messages embedded using *Doc2Vec* without being given the number of clusters, with the idea that messages in the same conversation would form a cluster. Note that message pairs in the training and validation data are not utilized in prediction for a fair comparison to all methods.

Table 3 shows the performance of conversation disentanglement. Note that "Oracle" represents the optimal performance for CISIR when all message pairs in identical conversations in $\boldsymbol{D}$ are correctly retrieved. Because pairs in $\boldsymbol{D}$ may not have enough coverage to connect all messages in a coversation, the optimal performance could be lower than 1.0. CISIR performs better than all baseline methods for all datasets, and achieves excellent performance in IRC, due in part to the high-performing similarity estimates from the first stage. Among the baseline methods, GTM performs relatively well on all datasets except for IRC. This is because messages are more frequently posted in the IRC dataset, thereby increasing the number of incorrect pairs in the constructed graph. Examining the graph constructed by GTM, there are only two connected components, indicating that many conversations were incorrectly combined; in contrast, CISIR may be exempt from error propagation because it only relies on top-ranked pairs. Doc2Vec is trained to predict words in a document in an unsupervised manner. Its lowest performance in the experiments may point out a need for supervised learning in the specific task of conversation disentanglement to tackle the variation in semantic patterns. Time and author contextual cues do help conversation disentanglement as seen in the results of Block-10 and Speaker. Both of these contexts are integrated into our model.

5 Conclusions

In this paper, we propose a novel framework for disentangling conversations, including similarity estimation for message pairs and conversation identification. In contrast to previous work, we assume that we do not need to select all message pairs in the first stage, thereby reducing computational time without sacrificing performance too much. To estimate conversation-level similarity, a Siamese Hierarchical Convolutional Neural Network, SHCNN, is proposed to minimize the estimation error as well as preserve both the low- and high-level semantics of messages. In the second stage, we developed the Conversation Identification by SImilarity Ranking, CISIR, algorithm, which exploits the assumption made in the first stage and identifies individual, entangled conversations with high-ranked message pairs. Extensive experiments conducted on four publicly available datasets show that SHCNN and CISIR outperform several existing approaches in both similarity estimation and conversation identification.

Acknowledgement

We would like to thank the anonymous reviewers for their helpful comments. The work was partially supported by NIH U01HG008488, NIH R01GM115833, NIH U54GM114833, and NSF IIS-1313606.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *OSDI'16*. volume 16, pages 265–283.

James Allan. 2002. Introduction to topic detection and tracking. *Topic detection and tracking* pages 1–16.

Hadi Amiri, Philip Resnik, Jordan Boyd-Graber, and Hal Daumé III. 2016. Learning text pair similarity with context-sensitive autoencoders. In *ACL'16*. ACL, pages 1882–1892.

Paul M Aoki, Margaret H Szymanski, Luke Plurkowski, James D Thornton, Allison Woodruff, and Weilie Yi. 2006. Where's the party in multi-party?: Analyzing the structure of small-group sociable talk. In *CSCW'06*. ACM, pages 393–402.

Erik Aumayr, Jeffrey Chan, and Conor Hayes. 2011. Reconstruction of threaded conversations in online discussion forums. In *ICWSM'11*. pages 26–33.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *TPAMI* 35(8):1798–1828.

Phil Blunsom, Edward Grefenstette, and Nal Kalchbrenner. 2014. A convolutional neural network for modelling sentences. In *ACL'14*. ACL, pages 655–665.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606* .

D Manning Christopher, Raghavan Prabhakar, and SCHÜTZE Hinrich. 2008. Introduction to information retrieval. *An Introduction To Information Retrieval* 151:177.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *ICLR'16*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR* 12:2493–2537.

Wenchao Du, Pascal Poupart, and Wei Xu. 2017. Discovering conversational dependencies between messages in dialogs. In *AAAI'17*. pages 4917–4918.

Micha Elsner and Eugene Charniak. 2008. You talking to me? a corpus and algorithm for conversation disentanglement. In *ACL'08*. ACL, pages 834–842.

Micha Elsner and Eugene Charniak. 2010. Disentangling chat. *Computational Linguistics* 36(3):389–409.

Micha Elsner and Eugene Charniak. 2011. Disentangling chat with local coherence models. In *ACL-HLT'11*. ACL, pages 1179–1189.

Brendan J Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *science* 315(5814):972–976.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

William A Hamilton, Oliver Garretson, and Andruid Kerne. 2014. Streaming on twitch: fostering participatory communities of play within live mixed media. In *CHI'14*. ACM, pages 1315–1324.

Jun Han and Claudio Moraga. 1995. The influence of the sigmoid function parameters on the speed of backpropagation learning. *From Natural to Artificial Neural Computation* pages 195–201.

Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP'14*. ACL.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR'16*.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *AAAI'15*. volume 333, pages 2267–2273.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *ICML'14*. pages 1188–1196.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521(7553):436–444.

Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *ICML'13*. volume 30.

Elijah Mayfield, David Adamson, and Carolyn Penstein Rosé. 2012. Hierarchical conversation structure prediction in multi-party chat. In *SIGDIAL'12*. ACL, pages 60–69.

Shikib Mehri and Giuseppe Carenini. 2017. Chat disentanglement: Identifying semantic reply relationships with random forests and recurrent neural networks. In *IJCNLP'17*. volume 1, pages 615–623.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS'13*. pages 3111–3119.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *AAAI'16*. pages 2786–2792.

Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2015. Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment analysis. In *EMNLP'15*. ACL, pages 2539–2544.

Zhaochun Ren, Jun Ma, Shuaiqiang Wang, and Yang Liu. 2011. Summarizing web forum threads based on a latent topic propagation process. In *CIKM'11*. ACM, pages 879–884.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR'15*. ACM, pages 373–382.

Dou Shen, Qiang Yang, Jian-Tao Sun, and Zheng Chen. 2006. Thread detection in dynamic text message streams. In *SIGIR'06*. ACM, pages 35–42.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR* 15(1):1929–1958.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL'14*. pages 1555–1565.

Hongning Wang, Chi Wang, ChengXiang Zhai, and Jiawei Han. 2011a. Learning online discussion structures by conditional random fields. In *SIGIR'11*. ACM, pages 435–444.

Li Wang, Marco Lui, Su Nam Kim, Joakim Nivre, and Timothy Baldwin. 2011b. Predicting thread discourse structure over technical web forums. In *EMNLP'11*. ACL, pages 13–25.

Lidan Wang and Douglas W Oard. 2009. Context-based message expansion for disentanglement of interleaved text conversations. In *NAACL'09*. ACL, pages 200–208.

Shuohang Wang and Jing Jiang. 2017. A compare-aggregate model for matching text sequences. In *ICLR'17*.

Christopher C Werry. 1996. Internet relay chat. *Computer-mediated communication: Linguistic, social and cross-cultural perspectives* pages 47–63.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *TACL* 4:259–272.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS'15*. pages 649–657.