TILM: a Neural Language Model with Evolving Topical Influence

Shubhra Kanti Karmaker Santu
and **Kalyan Veeramachaneni**
MIT LIDS
Cambridge, Massachusetts, USA
santu@mit.edu
kalyanv@mit.edu

ChengXiang Zhai
Department of Computer Science
University of Illinois Urbana Champaign
Urbana, Illinois, USA
czhai@Illinois.edu

Abstract

Content of text data are often influenced by contextual factors which often evolve over time (e.g., content of social media are often influenced by topics covered in the major news streams). Existing language models do not consider the influence of such related evolving topics, and thus are not optimal. In this paper, we propose to incorporate such topical-influence into a language model to both improve its accuracy and enable cross-stream analysis of topical influences. Specifically, we propose a novel language model called Topical Influence Language Model (TILM), which is a novel extension of a neural language model to capture the influences on the contents in one text stream by the evolving topics in another related (or possibly same) text stream. Experimental results on six different text stream data comprised of conference paper titles show that the incorporation of evolving topical influence into a language model is beneficial and TILM outperforms multiple baselines in a challenging task of text forecasting. In addition to serving as a language model, TILM further enables interesting analysis of topical influence among multiple text streams.

1 Introduction and Motivation

Language modeling plays a central role in many Natural Language Processing (NLP) tasks and applications. Neural language models have attracted much attention recently due to their superior performance (Dieng et al., 2016; Kiros et al., 2014; Kiddon et al., 2016; Ranzato et al., 2015; Devlin et al., 2018; Peters et al., 2018). One common limitation of the existing neural language models is that they cannot model the potential influence of related contextual factors on text content generation. However, as text data are produced by humans based on their observations of the real world, the content of text data are generally influenced by many contextual factors, and thus, it is necessary to model the influence of those contextual factors on the generation of text content in order to optimize language modeling. For example, the content of social media may be influenced by the popular topics in news stream; another example is that the content of research papers in one research community such as *Information Retrieval* are often influenced by the topics of research papers published in the same community in the past or research papers published in another related research community such as *Machine Learning* since the general algorithms developed in the latter may be applied to solve application problems in the former.

In this paper, we propose to incorporate such topical influence into a language model to both improve its accuracy and enable cross-stream analysis of topical influences. Specifically, we propose a novel language model called Topical Influence Language Model (TILM), which is a novel extension of a neural language model to capture the influence on the contents in one text stream by the evolving topics in another related (or possibly same) text stream. In other words, TILM is a recurrent neural network-based deep learning architecture that incorporates topical influence to model the generation of a dynamically evolving text stream. Since most text data have time stamps associated with them, which generally indicate the time when a text document was produced, text data can often be regarded as a sequence / stream of text objects ordered by their time stamps. TILM is designed to model the generation of such a text stream, i.e., the generation of text data conditioned on a given time stamp. We also assume that the distributions of words in the text data from two different timestamps are somewhat different, which allows us to capture the evolution of topics in the stream data.

TILM is comprised of three basic components. The first component is the *Sequence Generator*, which can be any current neural language model.

The second component is the *Topic Generator* which captures the trend of different topics of interest within the *influencing text stream*, which can be based on any topic models, e.g., LDA. Finally, the third component is a novel *Influence Generator*, which ensures that TILM can capture topical influence corresponding to the input timestamp and incorporate it into the generation process to ensure that the generated text is consistent with that particular timestamp from the perspective of topical influence. TILM combines these three essential components into a single unified model and learns all their optimal parameter values from a training text stream corpus with time stamps.

We conducted comprehensive experiments with six different sets of publication stream datasets to evaluate the effectiveness of TILM. These datasets contain titles of the papers published in different machine learning theory and applied machine learning conferences over 20 years, which were collected from the Open Academic Graph. Experimental results show that the incorporation of topical influence into a language model is beneficial and TILM outperforms multiple baselines by a clear margin in a challenging task of text forecasting. We found that effectively capturing the evolving topical influence is the key to improving the accuracy of language modeling. In addition to serving as a language model, TILM further enables interesting analysis of topical influence among multiple text streams.

2 Related Works

There has been a surge of research interest in the use of neural network (NN) architectures for language modeling and automatic text generation in recent years (Dieng et al., 2016; Kiros et al., 2014; Kiddon et al., 2016; Ranzato et al., 2015). The first NN-based text generator was proposed by Kukich (Kukich, 1987), although generation was done only at the phrase level. Recent advances in recurrent neural network-based language models (RNN-LM) have demonstrated the value of distributed representations and its power to model arbitrarily long dependencies (Mikolov et al., 2010, 2011). Sutskever et al. (Sutskever et al., 2011) introduced a simple variant of the RNN that can generate meaningful sentences by learning from a character-level corpus. Mao et.al. have demonstrated how Recurrent Neural Networks, specially, Long-Short-Term-Memory (LSTM) is effective in solving various text generation tasks (Mao et al., 2014). TopicRNN proposed by Dieng (Dieng et al., 2016) integrated the merits of RNNs and latent topic models to capture long-range semantic dependency. Recently, Generative Adversarial Nets (GANs) that use a discriminative model to guide the training of the generative model has shown promising results in automated text generation (Rajeswar et al., 2017; Lin et al., 2017; Che et al., 2017; Zhang et al., 2017). All these text generation techniques have also been vastly studied for generating summary for text corpora [see (Gambhir and Gupta, 2017) for a comprehensive survey]. Most recently, BERT (Devlin et al., 2018) and ELMo (Peters et al., 2018) have shown very promising performance. However, none of these existing methods can model topical influences from related contextual factors on generation of text data. While external topical influences have been studied in the context of search behavior modeling (Karmaker Santu et al., 2017, 2018), similar study has not been pursued yet for text generation modeling. Specifically, current text generation processes are static processes with no notion of time and thus, can not model dynamically evolving text stream data corresponding to evolving influences of related text streams, which the proposed TILM can do.

3 Topical Influence Language Model

We first briefly discuss some preliminaries and notations. Next, we present the three major components that are the building blocks of the proposed Topical Influence Language Model (TILM) and then present how TILM combines them into a single unified model.

3.1 Preliminaries and Notations

The goal task involves performing language modeling on each text stream within a set of text streams, $S = \{s_1, s_2, ..., s_m\}$, where, $|S| = m$. Each text stream consists of a set of tuples in the form of (document, timestamp) , e.g., $s_1 = \{(d_1, t_1), (d_2, t_2), ...\}$. Each document is a sequence of words, e.g., $d_1 = [x_1 x_2 ...]$, where, each $x_i \in V$ and V is the vocabulary set. For modeling purposes, we group the documents within each stream into different bins based on their corresponding timestamp. The time ranges $[t_{start}, t_{end}]$ used to create these bins are defined by the user and can be in varying granularity like year, month, day or seconds. Each bin is then assigned a discrete timestamp T and all the documents in that bin share the same timestamp, e.g., all paper published in different machine learning conferences during year 2016 are assigned the timestamp $T_{2016} = [Jan_{2016}, Dec_{2016}]$.

3.2 Sequence Generator

Sequence Generator is the basic component of TILM which generates the next word x_i in the sentence given $i-1$ previous words. Thus, *Sequence Generator* is essentially a language model which generates a probability distribution over a sequence of words that can be used to predict the next word in the sequence. Any sequence modeling framework, e.g., Hidden Markov Models, Recurrent Neural Networks etc. can work as a sequence generator. For the experiments described in this paper, we chose recurrent neural network with LSTM cells as the *Sequence Generator* due to its recent promising results obtained for language modeling tasks (Kiros et al., 2014; Kim et al., 2016). Given the previous i words, i.e., $x_{1:i}$, the recurrent neural network based language models compute the conditional probability for the next word $y_i = v$ for $v \in V$, the vocabulary set, by computing a hidden state h_i and passing it through a Softmax function:

$$P(y_i = v|x_{1:i}) \equiv P(y_i = v|h_i) \tag{1}$$

$$P(y_i|h_i) \propto \exp(W^{\Omega} h_i + B^{\Omega}) \tag{2}$$

$$h_i = \Omega(h_{i-1}, x_i) \tag{3}$$

Here, Ω can be a standard RNN cell or more complicated cell like LSTM, GRU etc and W and B are linear transformation coefficients. Output at step i, i.e., y_i is fed as input for step $i+1$, thus, $x_{i+1} = y_i$.

3.3 Topic Generator

The next component of TILM is the *Topic Generator*. The primary purpose of this component is to analyze different topics across a related text stream data (let us call it s) and compute the evolution of topic distributions within that stream over time. It takes all past text stream data of s as input and applies a probabilistic topic model to infer n (a user defined parameter) different topics, each represented with a unique distribution over the entire vocabulary. Again, many different topic models can be potentially used, but in our experiments, we chose LDA (Blei et al., 2003) as the *Topic Generator* since it has been the most popular topic modeling technique in the last decade. Once n topics are identified, the *Topic Generator* takes all text content from each discrete timestamp T (defined in section 3.1) separately and computes the distribution of n topics over each timestamp, which we denote by θ_T.

The *Topic Generator* also provides a subcomponent, i.e., *History Extractor*, which, given a particular timestamp T as input, retrieves the topic distributions of previous r (a user defined parameter) timestamps computed by LDA. We mathematically denote the output of *History Extractor* by $\theta_{T-r:T-1}$, where, $\theta_{i:j}$ denotes topic distributions from timestamp i to j augmented into a single vector. This means the cardinality of vector $\theta_{T-r:T-1}$ is $r \times n$. In the case of modeling influence from multiple related text streams and assuming we have m such streams, we can simply concatenate $\theta_{T-r:T-1}$ from each stream to create a single vector of dimension $r \times n \times m$.

3.4 Influence Generator

The *Influence Generator* is a pivotal component of TILM, which models the evolving topical influence during the text generation process. Given a particular timestamp T, we represent topical influence by a real valued vector (γ_T) of dimension K (another user defined parameter), which is essentially the output of *Influence Generator*. The input to the *Influence Generator* is the $r \times n$ dimensional vector of topic distributions from previous r timestamps of a related text stream, i.e., $\theta_{T-r:T-1}$ (assuming T as the current timestamp). Thus, *Influence Generator* essentially maps a $r \times n$ dimensional topic vector to a K dimensional influence vector. Although any function that can perform this mapping can resemble as Influence Generator, we chose a feed-forward neural network for TILM due to its capability of approximating a wide family of functions. Without loss of generality, we used ReLU activation units in the hidden layers. Once the influence vector (γ_T) is computed, it is then injected as a bias into the *Sequence Generator* when generating the next word (x_i) in the sequence (More details in section 3.5).

Here, we assume that the influence vector γ_T corresponding to current timestamp T, can be approximated from the historical topic distribution $\theta_{T-r:T-1}$. This assumption is reasonable, because most text stream data do not evolve dramatically over night, rather their topical shift happens quite gradually. Take for example some particular research community like SIGKDD. The topic distribution in papers published in a particular year is unlikely dramatically different from the previous two years, rather they are somewhat correlated.

Mathematically, let θ_T denote the topic distribution for the generated text at timestamp T, then the function of *Influence Generator* is expressed as follows ($\|$ means the concatenation operation):

$$\theta_{T-r:T-1} = \theta_{T-r}\|\theta_{T-r+1}\|....\|\theta_{T-1} \quad (4)$$

$$\begin{aligned}\gamma_T &= \Gamma(\theta_{T-r:T-1})\\ &= W_2^\Gamma \cdot \left[ReLU(W_1^\Gamma \cdot \theta_{T-r:T-1} + B_1^\Gamma)\right] + B_2^\Gamma\end{aligned} \quad (5)$$

3.5 TILM as a Unified Model

Now that we have presented the three building blocks of TILM, this section presents how these different components interact with each other and work as a unified architecture to model stream text data by capturing the fluctuations of topical influence over time. The process can be thought of as generating text corresponding to a particular timestamp T. Thus, the whole process starts with a timestamp T as input and the start-of-sentence marker (let's call it #) as the sequence generated so far. The next task is to generate one word at a time iteratively until the end-of-sentence marker is generated (let's call it *). The exact process of generating the next word y_i in the sequence is demonstrated in Figure 1.

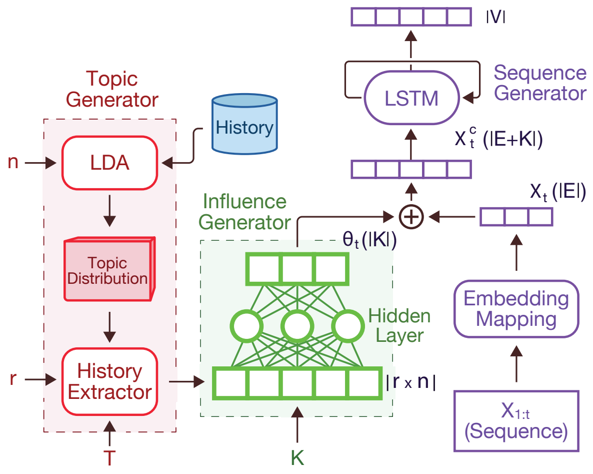


Figure 1: TILM architecture: Topic Generator (Red), Influence Generator (Green) and Sequence Generator (Purple)

The first step of generation process is to infer n different topics from a related (possibly same) historical text stream and compute the topic distributions for each unique timestamp observed in the training data. Next, given a particular timestamp T as input, the *History Extractor* Module extracts the historical topic distributions corresponding to previous r timestamps and concatenates them to generate a vector representation of the history, i.e., $\theta_{T-r:T-1}$ of dimension $r \times n$ (see the previous subsection for details). $\theta_{T-r:T-1}$ is then passed through *Influence Generator* Γ which outputs the K dimensional influence vector γ_T. For a particular timestamp T, γ_T is fixed and can be re-used for any text generation task tied to the timestamp T. The bottom middle section (Green Color) of Figure 1 shows the feed-forward neural network of *Influence Generator*.

The next trick in TILM is to concatenate the influence vector (γ_T) with the vector representation of each word in the sequence generated so far. This means, for each word in $\{x_1, x_2, ..., x_i\}$, γ_T is concatenated to each of their vector representations to create an augmented representation $\{x_1^C, x_2^C, ..., x_i^C\}$, i.e., while generating the next word $x_{i+1} = y_i$ in the sequence, all the previous words in the sequence share the same topical influence represented by vector γ_T. This augmented representation essentially allows TILM to capture the dynamic nature of text stream data as the influence vector injects evolving topical influence into the generation process. Finally, the augmented representations $\{x_1^C, x_2^C, ..., x_i^C\}$ are fed into the recurrent neural network model to compute a hidden state h_i . The final output vector Y is computed by applying a linear transformation on h_i. Note that, vector Y is a real-valued vector. We apply a Softmax function on Y to convert it into a valid probability distribution, sampling from which, the next word in the sequence is generated. The mathematical formulas behind the entire generation process is summarized below:

$$Y = W^\Omega \cdot h_i + B^\Omega, \quad (6)$$

$$h_i = \Omega(h_{i-1}, x_i^C), \quad (7)$$

$$x_i^C = x_i \| \gamma_T. \quad (8)$$

Thus, Y can be written as follows:

$$Y = W^\Omega \cdot \Omega(h_{i-1}, x_i \| \gamma_T) + B^\Omega. \quad (9)$$

Here, γ_T is obtained as follows:

$$\gamma_T = W_2^\Gamma \cdot \left[ReLU(W_1^\Gamma \cdot \theta_{T-r:T-1} + B_1^\Gamma)\right] + B_2^\Gamma. \quad (10)$$

Here, $\theta_{T-r:T-1}$ is the concatenation of topic distribution vector from previous r timestamps.

Finally, we apply a Softmax function on the output vector Y to convert it into a valid probability distribution $P(y_i|h_i)$, as follows:

$$P(y_i = v|h_i) = \frac{\exp(Y_i)}{\sum_{j=1}^{|V|} \exp(Y_j)}. \qquad (11)$$

The next word $y_i = x_{i+1}$ in the sequence is generated by sampling from this conditional distribution. TILM repeats this whole process multiple times to generate new words in the sentence until a end-of-sentence marker is generated. The whole process is summarized in Algorithm 1, which describes the generation of a single sentence by TILM for a particular timestamp T.

Algorithm 1: Topical Influence Language Model (TILM)

```
1  Process TILM ($T, \Theta, \Gamma, \Omega, r, n, K, E$);
   Input  : $T$: discrete timestamp
            $\Theta$: Topic Generator ($n$: number of topics)
            $\Gamma$: Influence Generator ($K$: cardinality of influence vector)
            $\Omega$: Sequence Generator ($E$: cardinality of word vector)
            $r$: History Window
   Output: Generated sentence $X$ corresponding to $T$
2  $x \leftarrow$ {start of sentence marker}
3  $\theta_{T-r:T-1} \leftarrow \Theta(\theta_{T-r})||\Theta(\theta_{T-r+1})||...||\Theta(\theta_{T-1})$, Topic History
4  $\gamma_T \leftarrow \Gamma(\theta_{T-r:T-1})$, Generate Influence Vector for timestamp $T$
5  $i \leftarrow 1$
6  repeat
7      for $j \leftarrow 1$ to $i$ do
8          $x_j^C \leftarrow x_j||\gamma_T$, where, $||$ is concatenation operation
9      end
10     compute $h_i \leftarrow \Omega(x_{1:i}^C)$ by applying $\Omega$ recursively
11     Draw word $y_i \sim P(y_i|h_i)$, where $P(y_i|h_i) \propto \exp(W^\Omega h_i + B^\Omega)$
12     $x \leftarrow x \cup \{y_i\}$
13     $i \leftarrow i + 1$
14 until end of sentence marker is generated;
15 return $x$
```

3.6 Estimation of TILM parameters

In this section, we present the estimation techniques for the optimal values of TILM model parameters. Close observation of Equation 6-10 reveals that TILM contains the following set of parameters:

$$\boldsymbol{W} = \{W^\Omega, B^\Omega, W_1^\Gamma, B_1^\Gamma, W_2^\Gamma, B_2^\Gamma\} \qquad (12)$$

We find the optimal values for the parameter set $\boldsymbol{W}$ by maximizing the log-likelihood of the training text stream data. The optimization problem thus can be written as follows:

$$\boldsymbol{W}^* = \underset{W}{argmax} \quad \log L(x_1 x_2 ... x_n|W) \qquad (13)$$

As maximizing the log-likelihood is the same as minimizing the negative of the log-likelihood function and as we know the exact word which comes next in the sequence during the training process, our optimization problem boils down to minimizing the softmax cross entropy with logits between the conditional distribution $P(y_i|h_i)$ and the one-hot encoding of the actual word that appears next in the training data. Softmax Cross Entropy with logits essentially measures the probability error in discrete classification tasks in which the classes are mutually exclusive. Thus,

$$\boldsymbol{W}^* = \underset{W}{argmin} \{-\log L(x_1 x_2 ... x_n|W)\} \qquad (14)$$

$$= \underset{W}{argmin} \left\{ -\sum_{i=1}^{N} \sum_{v \in V} I(x_i, v) \cdot \log P(x_i = v|h_i(W)) \right\} \qquad (15)$$

Here, N is the total number of words in the training data. $I(p, q)$ is an indicator function that returns 1 if $p = q$ and 0 otherwise.

We used back-propagation to learn the weights of the network connection edges of TILM. Specifically, we used Adaptive Moment Estimation, which is a popular stochastic gradient descent technique and commonly known as Adam Optimizer, to compute the gradient for minimizing our objective function in Equation 15. Adam Optimizer is an update to the RMSProp (Hinton et al., 2012), which is another popular optimizer. For more details, refer to (Kingma and Ba, 2014).

4 Experimental Design

4.1 Dataset

We experimented with six different sets of publication title stream data to evaluate the performance of TILM. These datasets were collected from the Open Academic Graph[1](Tang et al., 2008; Sinha et al., 2015). Here, we focused on studying how the paper titles published by different machine learning related conferences evolved over time. As community, we considered both the core machine learning community, e.g., NIPS, ICML as well as research communities that apply a fair share of machine learning, e.g, KDD, SIGIR. Specifically, we considered all the titles of papers published during the years 1996-2015 by the following six conference venues: NIPS, CVPR, ICML, KDD, SIGIR and WWW. For these datasets, the discrete timestamp corresponds to a particular year.

[1]https://www.openacademic.ai/oag/

Conf.	# of Titles	Title/Year	Total Words	Words/Title
KDD	5,499	274.95	49,980	9.08
NIPS	6,229	311.45	49,792	7.99
SIGIR	3,994	199.7	34,892	8.73
ICML	4,106	205.3	34,159	8.32
WWW	5,701	285.05	50,253	8.82
CVPR	10,121	506.05	90,890	8.98

Table 1: Dataset Summary

Each row in these datasets consists of a tuple <timestamp, paper-title> and altogether they contain $35,650$ paper titles in total. Again, each paper title can contribute multiple instances for predicting the next word which resulted in $309,966$ total instances (see Table 1).

4.2 Evaluation Roadmap

For a proper evaluation of TILM, we need a goal task which is both time-sensitive and requires influence modeling. To address this challenge, we evaluate TILM using a text forecasting task, where we attempt to predict the text content at future timestamps in a text stream based on the past data from related (including the same) streams.

As the setup of such a text forecasting task is essentially similar to summarizing *future* text data, we use two popular evaluation metrics from the literature of text summarization, i.e., BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004), where a score is generated by comparing the automatically generated text against some reference text written by humans. However, neither BLEU nor ROUGE considers the notion of time, thus we need a time-sensitive customization of both BLEU and ROUGE. The simplest way to do this is to compare a TILM generated text for timestamp T against the original text corresponding to the same timestamp T. This means, when TILM is asked to generate a paper title relevant to timestamp T, the ground truth paper title against which the generated title is compared must also correspond to the timestamp T. Another challenge is to match the generated title against multiple independent publication titles corresponding to timestamp T. We address it by adopting a simple greedy approach where the TILM-generated title is matched against each ground truth title corresponding to the timestamp T and paired with the most similar one in terms of BLEU or ROUGE score. The matched ground truth title is then removed from the corpus so that the next generated title cannot match with the previously matched title again. This ensures that TILM is generating a diverse set of titles rather than just memorizing one single title from each timestamp T. This way, we can use TILM to generate multiple sentences (titles) for a particular timestamp T and then average the scores of all generated sentences to get an evaluation score corresponding to timestamp T. This whole computation process is presented in Algorithm 2, where we demonstrate the case for time-sensitive BLEU score. The case for ROUGE is exactly similar. Finally, these average scores across different timestamps can be further averaged to compute the overall forecasting score[2].

Algorithm 2: Time aware BLEU score computation

```
1  Time aware BLEU $(T, G, R)$;
   Input  : $T$: discrete timestamp
            $G$: Generated Text set
            $R$: Reference Text set
   Output: Time sensitive BLEU
2  $score \leftarrow 0$
3  $|G| \leftarrow$ number of sentences in $G$
4  for each sentence $g$ in $G$ do
5      for each sentence $r$ in $R$ do
6          compute $BLEU(g, r)$
7      end
8      $r^* = \underset{r}{argmax} \; BLEU(g, r)$
9      $score \leftarrow score + BLEU(g, r^*)$
10     $R \leftarrow R - \{r^*\}$
11 end
12 return $\frac{score}{|G|}$
```

4.3 Baseline Methods

The main questions we want to answer in our experiments are whether the incorporation of topical influence (from a related stream) is beneficial for language modeling and whether the specific configuration of TILM we described earlier is an effective way to capture evolving topical influence. To answer the second question, we compare TILM with two baselines which are both variants of TILM but with different ways to capture influence. The first one is called *RILSTM* which is identical to TILM except that the influence vector of RILSTM is generated randomly as opposed to generating it by the *Influence Generator* of TILM. The second baseline is called *IILSTM* where we do not inject the influence vector as a bias into

[2]All the codes and evaluation scripts for experimentation can be found at the following link: (`https://bitbucket.org/karmake2/tilm/src/master/`

Acronym	Details	Nature
Bigram	Bigram Language Model	Static
LSTM	Long short-term memory	Static
RILSTM	LSTM with Random Influence	Dynamic
IILSTM	Sampling from Joint LSTM-Influence Distribution	Dynamic
TILM	Topical Influence LM	Dynamic

Table 2: Baselines for Quantitative Comparison.

the vector representation of words, rather, the *Influence Generator* directly computes a probability distribution for sampling the next word and this probability is multiplied with the probability computed independently by LSTM. To answer the first question, i.e., to verify the benefit of modeling the evolution of topical influence, we also compared TILM against two baselines representing static models: simple bigram language model and Long-Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997; Gers et al., 1999). Table 2 contains the summary of these baseline algorithms along with TILM.

5 Results

This section presents both quantitative and qualitative evaluation results for TILM. For all the results reported in this section, TILM used the following parameter settings: r was set to 3, for both *Sequence Generator* and *Influence Generator*, the number of hidden units was empirically set to 256, K (dimension of influence vector) was set to 15, batch size was set to 2000 instances and the learning rate was set to 0.01. For the *Topic Generator*, n (number of topics) was set to 15, θ_T was computed from target text stream itself and LDA was run using $\alpha = 0.1$ and $\beta = 0.05$.

5.1 Quantitative Evaluation

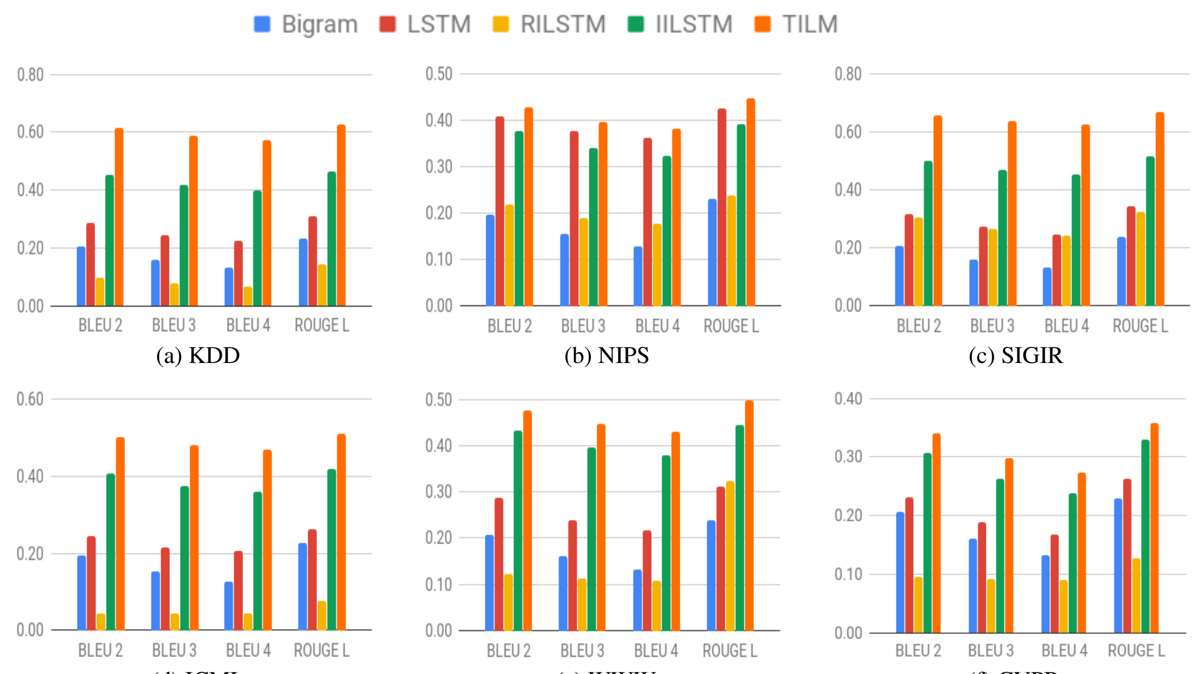
Figure 2 provides the summary of results for comparing TILM against multiple baseline methods. Close examination of Figure 2 reveals that TILM outperforms all other baselines by a clear margin for all six datasets. For example, BLEU-4 score obtained by TILM on KDD Dataset is 0.57, while LSTM obtained only a score of 0.22. ROUGE-L score obtained by TILM is 0.63, while it is 0.31 for LSTM. This clearly indicates that TILM can indeed capture the temporal evolution of KDD paper titles over time and given a input timestamp T, can generate text relevant to T. Also note that, RILSTM performs significantly worse compared to LSTM for most datasets which implies that the influence vector plays the key role in helping TILM capture the evolution of the text stream. It is also noteworthy that IILSTM is the second best performing method which confirms that injecting influence vector as a bias into the word representation works better than using the Joint LSTM-Influence distribution obtained by simply multiplying influence probabilities with LSTM probabilities.

To get more insights into the performance of TILM, we plot the timestamp-wise performance of all compared methods for KDD Dataset (BLEU 4) and WWW Dataset (ROUGE L) in Figure 3 [other plots are similar and omitted due to lack of space]. A general inspection of Figure 3 also demonstrates the superiority of TILM for the text forecasting task, where, for any performance metric, TILM obtains the best score across different timestamps for most of the cases.

External Influence: So far, we have only considered the influence of the community for which the text generation is targeted towards. However, other related communities also pose indirect influence on the text content generated within the target community. For example, a shift in the interest of theoretical machine learning conferences like ICML often influences the research directions pursued by more applied conferences like KDD or WWW. To test this hypothesis, we conducted a series of experiments where instead of using the influence of the target community (e.g., KDD, WWW) itself, we computed the influence vector from the historical topic distribution of a core machine learning community (e.g. ICML, NIPS). We call this approach TILM-EI where EI means external influence. We conducted another set of experiments where we computed influence vectors from both the target community and a related external community and injected both influence vectors into the TILM process. We call this approach TILM-CI where CI stands for combined influence. Two sample results from these experiments are shown in Figure 4, i.e., influence of ICML on KDD and WWW, respectively. Experiments results suggest that, although TILM-EI is not always better than TILM itself, TILM-CI outperforms basic TILM as TILM-CI combines both internal and external influence.

5.2 Qualitative Evaluation

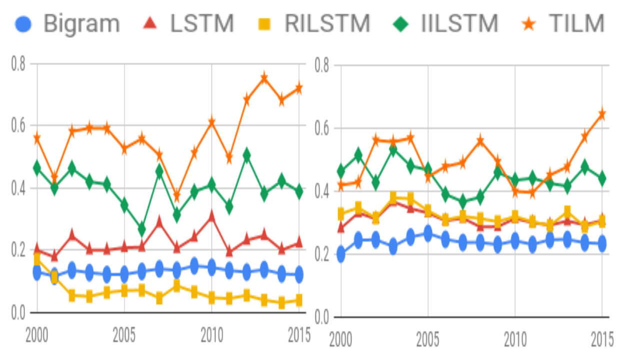
In this section, we present qualitative results to show the great potentials of TILM. We first ran LDA on paper titles from SIGIR and KDD over the year range 2000-2015. Number of topics was



(a) KDD (b) NIPS (c) SIGIR

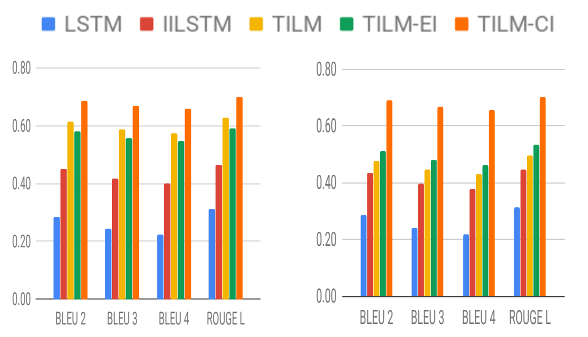
(d) ICML (e) WWW (f) CVPR

Figure 2: Comparison of TILM against baselines for text forecasting.



(a) KDD: BLEU 4 (b) WWW: ROUGE L

Figure 3: Year-Wise Performance distribution of TILM against different baseline text generation techniques



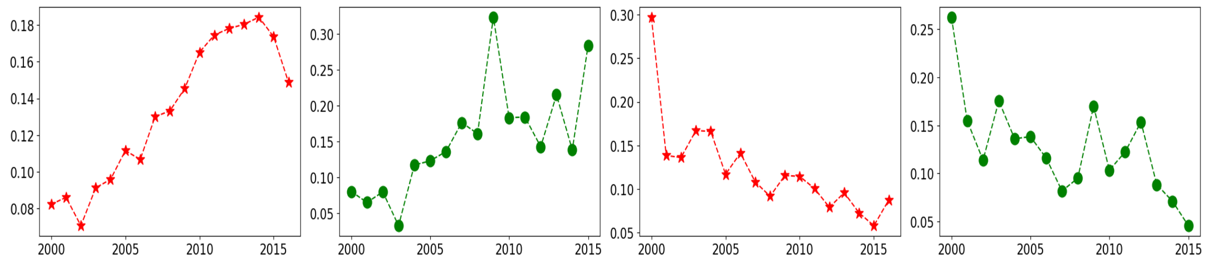
(a) ICML influencing KDD (b) ICML influencing WWW

Figure 4: Results of adding external influence into TILM for text forecasting task.

Topic	**Top Keywords**
Optimization	matrix, gradient, sparse, convex, stochastic
Search Relevance	information, retrieval, search, index, document
Rule Mining	rule, discovery, association, pattern, mine
Social Networks	social, network, recommender, community, topic
SVM Classifiers	supervised, learning, support, vector, machine
User Behavior Modeling	log, behavior, personalization, click, feedback

Table 3: Sample topics Extracted from KDD and SIGIR for year range [1995-2015] Using LDA

set to 15. Table 3 shows six example topics along with top 5 keywords for each topic. Next, we did some topic trend analysis for SIGIR conference in Figure 5. For SIGIR, we considered two topics, i.e., *User Behavior Modeling* [Figure 5a, 5b] and *Search and Relevance* [Figure 5c, 5d]. Beside plotting the original topic-distribution trend computed from the real conference proceedings (Figures in red color), we also plot the simulated topic-distribution trend computed from the text generated by TILM (Figures in green color). Close observation of Figure 5 confirms that TILM can indeed generate sentences aligned with the evolution of the text stream corresponding to evolving topical influence. For example, Figure 5a shows that research interest towards *User Behavior Modeling* grew significantly within SIGIR community in the past ten years, which is also nicely reflected in the text generated by TILM [Figure 5b]. On the other hand, research on Search and relevance almost matured after 2008 within the SIGIR community [Figure 5c], which has also been captured



(a) $SIGIR$(**Original**): User Behavior Modeling (b) $SIGIR$(**Generated**): User Behavior Modeling (c) $SIGIR$(**Original**): Search & Relevance (d) $SIGIR$(**Generated**): Search & Relevance

Figure 5: Topic trend analysis to demonstrate how TILM captures evolution of SIGIR research paper titles.

year	**#**	**Sample Generated Title**
2000	1	discovering in hierarchical rules using lexical knowledge
-	2	data mining criteria for tree based regression and classification
2002	3	mining frequent class sets in spatial databases
2007 -	1	a framework for community identification in dynamic social networks
2009	2	learning preferences of new users in recommender systems
	3	data mining for intrusion detection from outliers
2012 -	1	deep model based transfer and multi task learning for biological image analysis
2015	2	a bayesian framework for estimating properties of information network
	3	active learning for sparse bayesian classification

Table 4: Sample titles generated by TILM for conference KDD across different year ranges

effectively by TILM and apparent from the decaying trend of Figure 5d.

Finally, Table 4 presents some sample paper titles generated by TILM for different time ranges targeted towards KDD community. A closer look into Table 4 reveals that TILM can generate syntactically correct, semantically coherent and time-sensitive evolving text. It is worth mentioning that, TILM did not store any paper-to-year mapping information. Table 4 also nicely captures the interest shift within KDD community over the years. For example, paper titles generated for year range 2000-2002 include topics like rule mining and tree based classifications, while paper titles generated for year range 2012-2015 include topics like deep learning and active learning, which is consistent with our expectation.

6 Conclusion

We studied how to improve neural language models by incorporating topical influence from contextual factors and proposed a novel Topical Influence Language Model (TILM), which includes a novel extension of a basic neural language model by incorporating both a topic generator (based on topic modeling) and a neural network-based influence modeling component, leading to a general architecture with three major components: *Sequence Generator*, *Topic Generator* and *Influence Generator*. We quantitatively evaluated TILM using a text forecasting task on six publication stream datasets and demonstrated that it is beneficial to incorporate topical influence in language modeling and TILM outperforms multiple baseline methods by a significant margin. As a novel language model, TILM allows for leveraging related text streams to improve accuracy of language modeling of a target stream, thus potentially helping improve many applications where language models are applied. We also show that TILM is able to generate well-structured, meaningful text content corresponding to future time stamps, thus potentially allowing us to predict topical trends in the future, which would be useful for optimizing decision making. Moreover, we also showed the potential that TILM can be used as a tool to compare influences of different external streams on a particular target stream, thus facilitating cross-stream influence analysis. While it is not the focus of this paper, such analysis clearly opens up an interesting direction for analyzing multiple text streams to understand their influences in a topic-specific way, a highly promising direction for future research.

7 Acknowledgement

We sincerely thank Hao Ma, Bin Bi, Kuansan Wang and Jiawei Han for their valuable feedback during the initial phase of generating the idea for this paper. We also thank the three anonymous reviews for their unanimous recognition of the importance of this new research direction which would help promoting this direction to a greater extent.

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Tong Che, Yanran Li, Ruixiang Zhang, R. Devon Hjelm, Wenjie Li, Yangqiu Song, and Yoshua Bengio. 2017. Maximum-likelihood augmented discrete generative adversarial networks. *CoRR*, abs/1702.07983.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Adji B Dieng, Chong Wang, Jianfeng Gao, and John Paisley. 2016. Topicrnn: A recurrent neural network with long-range semantic dependency. *arXiv preprint arXiv:1611.01702*.

Mahak Gambhir and Vishal Gupta. 2017. Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66.

Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 1999. Learning to forget: Continual prediction with lstm.

G Hinton, N Srivastava, and K Swersky. 2012. Rmsprop: Divide the gradient by a running average of its recent magnitude. *Neural networks for machine learning, Coursera lecture 6e*.

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

Shubhra Kanti Karmaker Santu, Liangda Li, Yi Chang, and ChengXiang Zhai. 2018. Jim: Joint influence modeling for collective search behavior. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 637–646. ACM.

Shubhra Kanti Karmaker Santu, Liangda Li, Dae Hoon Park, Yi Chang, and Chengxiang Zhai. 2017. Modeling the influence of popular trending events on user search behavior. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 535–544. International World Wide Web Conferences Steering Committee.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 329–339.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *AAAI*, pages 2741–2749.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. 2014. Multimodal neural language models. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 595–603.

Karen Kukich. 1987. *Where do Phrases Come from: Some Preliminary Experiments in Connectionist Phrase Generation*, pages 405–421. Springer Netherlands, Dordrecht.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain.

Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, and Ming-Ting Sun. 2017. Adversarial ranking for language generation. *CoRR*, abs/1705.11001.

Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. 2014. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632*.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048.

Tomas Mikolov, Stefan Kombrink, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011, May 22-27, 2011, Prague Congress Center, Prague, Czech Republic*, pages 5528–5531.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Sai Rajeswar, Sandeep Subramanian, Francis Dutil, Christopher Joseph Pal, and Aaron C. Courville. 2017. Adversarial generation of natural language. *CoRR*, abs/1705.10929.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-june Paul Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th* *international conference on world wide web*, pages 243–246. ACM.

Ilya Sutskever, James Martens, and Geoffrey E. Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 1017–1024.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998. ACM.

Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. 2017. Adversarial feature matching for text generation. *arXiv preprint arXiv:1706.03850*.