# TopicRNN: A Recurrent Neural Network with Long-Range Semantic Dependency

Adji B. Dieng[†], Chong Wang[°], Jianfeng Gao[°], and John Paisley[†]

[†]Columbia University, [°]Microsoft Research

## Summary and Contributions


- Neural network-based language models have achieved state of the art results on many NLP tasks.
- One difficulty is to capture long-range dependencies.
- We use latent topics to capture semantic and RNNs to capture syntax in a simple and end-to-end trainable architecture: TopicRNN.
- More specifically, we use topic features learned by an encoder as additional bias to the softmax layer of an RNN.
- We use a binary switching variable determined by the hidden layer of the RNN to decide whether we predict a stop word or a content word. The topic feature is useful only for predicting content words.
- TopicRNN is trained using the Adam algorithm on the evidence lower bound.
- TopicRNN shows better generalization abilities on the Penn Treebank and achieves SOTA-comparable sentiment classification error rate on the IMDB.


## Language Modeling

- A language model is a distribution over a sequence of words:

$$p(w_1, \ldots, w_T) = p(w_1) \prod_{t=2}^{T} p(w_t | w_{1:t-1}).$$

- RNN-based language models [3] define the conditional probability of each word $w_t$ given all the previous words $w_{1:t-1}$ through their hidden state via a softmax.
- However, RNNs face difficulties remembering very long-range dependencies.
- Modeling long-range dependencies in language is a research challenge.
- The intuition below motivates our work in this area.

## Intuition: Syntax is local, semantic is global

"*The U.S. presidential race isn't only drawing attention and controversy in the United States – it's being closely watched across the globe. But what does the rest of the world think about a campaign that has already thrown up one surprise after another? CNN asked 10 journalists for their take on the race so far, and what their country might be hoping for in America's next President* "

## TopicRNN: Capturing semantic via latent topics and syntax via RNNs

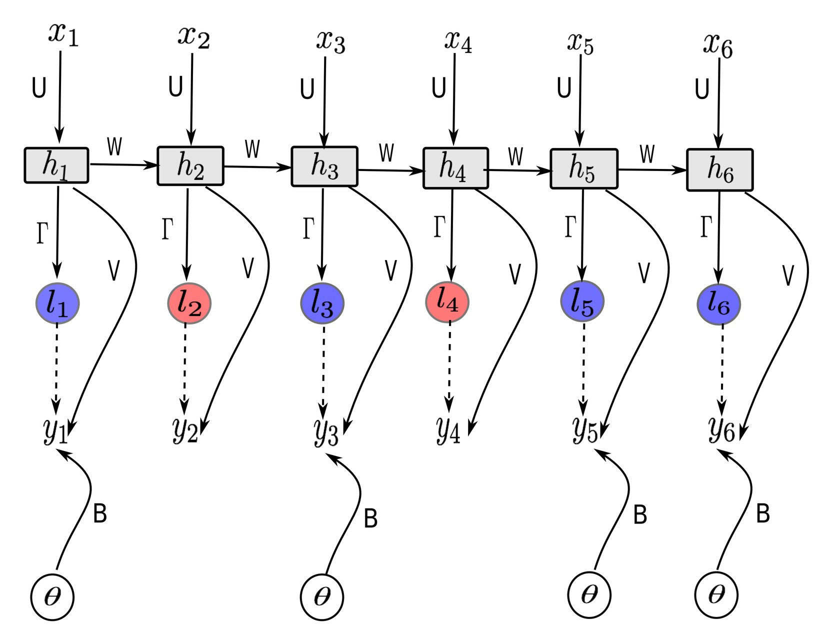



Figure 1: The unrolled TopicRNN architecture: $x_1, \ldots, x_6$ are words in the document, $h_t$ is the state of the RNN at time step $t$, $x_i \equiv y_{i-1}$, $l_1, \ldots, l_6$ are stop word indicators, and $\theta$ is the latent representation of the input document and is unshaded by convention.

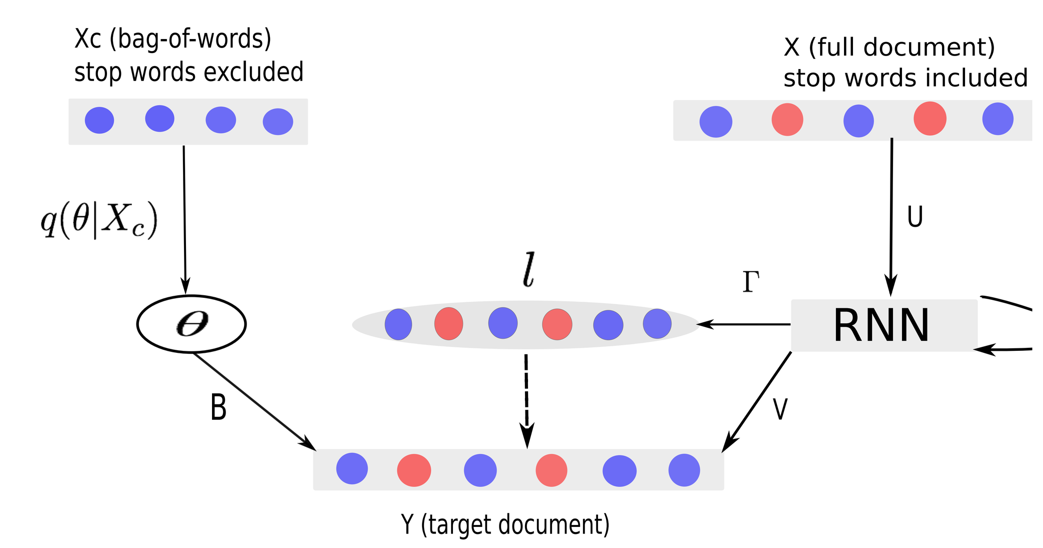


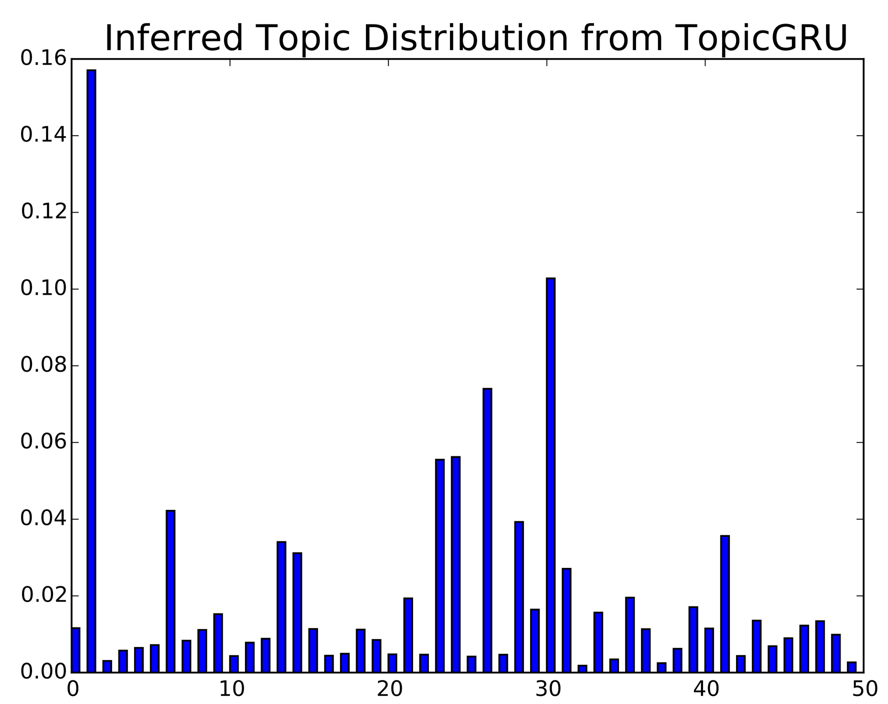

Figure 2: The TopicRNN model architecture in its compact forn binary vector that indicates whether each word in the input document is a stop word or not. Here red indicates stop words and blue indicates content words.

$$\mathcal{L}(\Theta) \triangleq -\mathbb{E}_{q(\theta|X_c, W_c)}\left(\sum_{t=1}^{T} \log p(y_t | h_t, l_t, \theta) + \log p(l_t | h_t) + \log p(\theta) - \log q(\theta | X_c, W_c)\right) \leq \log p(y_{1:T}, l_{1:T} | h_t, \Theta).$$

$$q(\theta | X_c, W_c) = N(\theta; \mu(X_c), \mathrm{diag}(\sigma^2(X_c))).$$

$$p(y_t | y_{1:t-1}) \approx \sum_{l_t} p(y_t | h_t, \hat{\theta}, l_t) p(l_t | h_t).$$

## Topic Discovery and Word Prediction on Penn Treebank



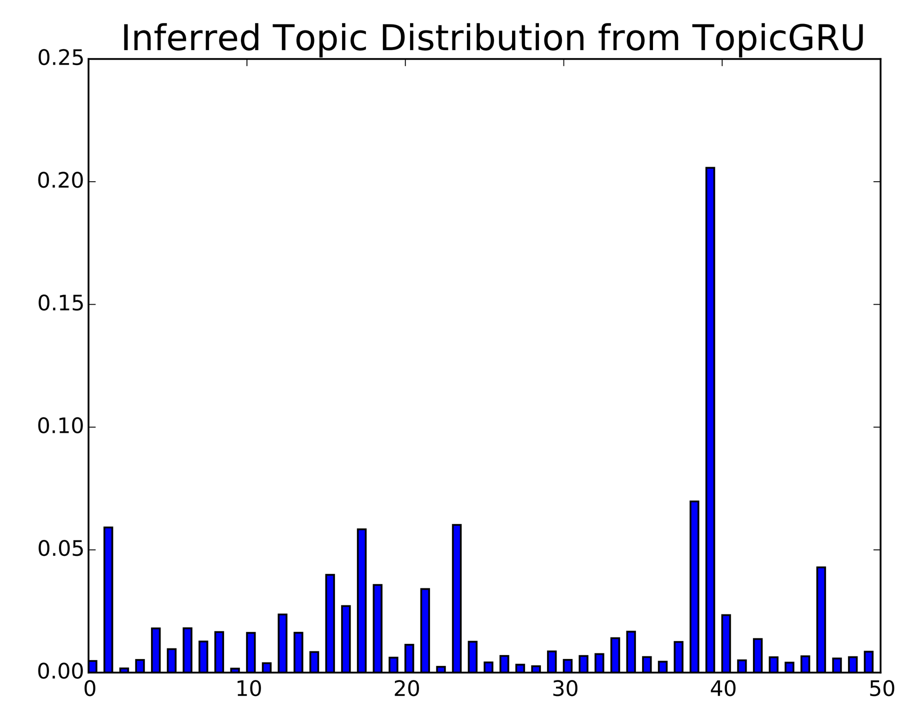




Table 1: Five Topics from the TopicRNN Model. More results can be found on the paper.

| Law | Company | Parties | Trading | Cars |
|---|---|---|---|---|
| law | spending | democratic | stock | gm |
| lawyers | sales | republicans | sp | auto |
| judge | advertising | gop | price | ford |
| rights | employees | republican | investor | jaguar |
| attorney | state | senate | standard | car |
| court | taxes | oakland | chairman | cars |

Table 2: Test perplexity on the Penn TreeBank for different number of hidden units. **1 TopicGRU with 100 units performs better than 2 stacked LSTM of 200 units each (112.4 vs 115.9)**.

| 100 Neurons | Valid | Test |
|---|---|---|
| RNN (no features) | 150.1 | 142.1 |
| RNN (LDA features) | 132.3 | 126.4 |
| TopicRNN | 128.5 | 122.3 |
| TopicLSTM | 126.0 | 118.1 |
| TopicGRU | 118.3 | **112.4** |

## Sentiment Analysis on IMDB

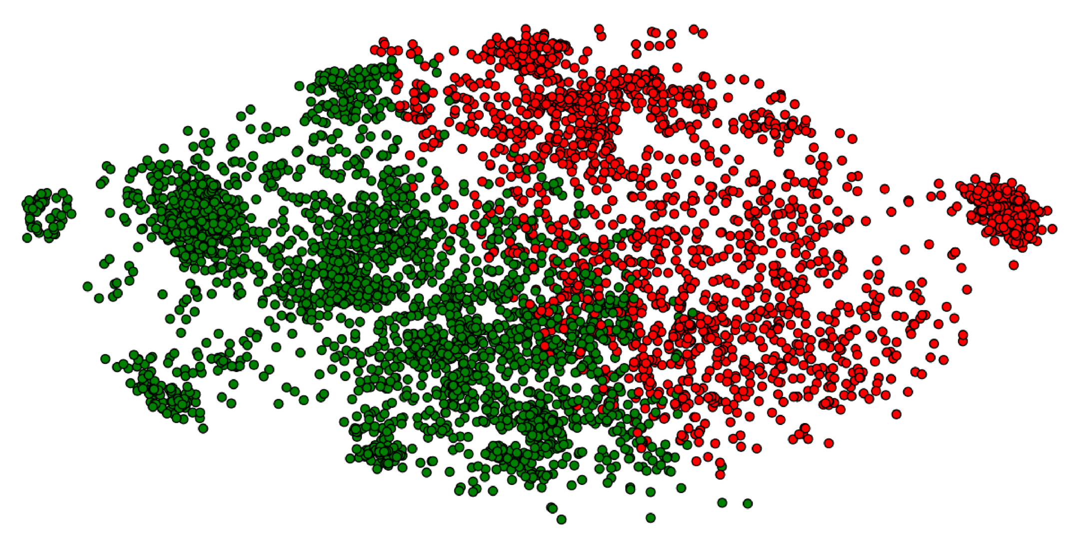


Figure 3: TopicRNN can be used as **unsupervised feature extractor**. Illustrated are review features discovered by TopicRNN, clustered via k-means, and projected via PCA for illustration.

Table 3: Classification error rate on IMDB 100k dataset. TopicRNN was used as unsupervised feature extractor. These review features were then used for classification. This achieved a **SOTA-comparable error rate**.

| Model | Reported Classification Error rate |
|---|---|
| BoW (bnc) (Maas et al., 2011) | 12.20% |
| BoW ($b\Delta$ tć) (Maas et al., 2011) | 11.77% |
| LDA (Maas et al., 2011) | 32.58% |
| Full + BoW (Maas et al., 2011) | 11.67% |
| Full + Unlabelled + BoW (Maas et al., 2011) | 11.11% |
| WRRBM (Dahl et al., 2012) | 12.58% |
| WRRBM + BoW (bnc) (Dahl et al., 2012) | 10.77% |
| MNB-uni (Wang & Manning, 2012) | 16.45% |
| MNB-bi (Wang & Manning, 2012) | 13.41% |
| SVM-uni (Wang & Manning, 2012) | 13.05% |
| SVM-bi (Wang & Manning, 2012) | 10.84% |
| NBSVM-uni (Wang & Manning, 2012) | 11.71% |
| seq2-bown-CNN (Johnson & Zhang, 2014) | 14.70% |
| NBSVM-bi (Wang & Manning, 2012) | 8.78% |
| Paragraph Vector (Le & Mikolov, 2014) | 7.42% |
| SA-LSTM with joint training (Dai & Le, 2015) | 14.70% |
| LSTM with tuning and dropout (Dai & Le, 2015) | 13.50% |
| LSTM initialized with word2vec embeddings (Dai & Le, 2015) | 10.00% |
| SA-LSTM with linear gain (Dai & Le, 2015) | 9.17% |
| LM-TM (Dai & Le, 2015) | 7.64% |
| SA-LSTM (Dai & Le, 2015) | 7.24% |
| **Virtual Adversarial (Miyato et al. 2016)** | **5.91%** |
| **TopicRNN** | **6.28%** |

## References


[1] David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[2] Yishu Miao, Lei Yu, and Phil Blunsom. Neural variational inference for text processing. *arXiv preprint arXiv:1511.06038*, 2015.

[3] Tomas Mikolov and Geoffrey Zweig. Context dependent recurrent neural network language model. In *SLT*, pages 234–239, 2012.