# BRiSK: Timeseries recommendations on the arXiv

DL, JA

January 22, 2016

Inference is expensive, especially in Bayesian timeseries models with long-term dependencies such as factorial hidden markov models. Recently, variational autoencoder methods (Kingma and Welling, 2014) have emerged to enable efficient inference of complex LSTM-based timeseries models (Bowman et al., 2015).

We apply such frameworks to a timeseries recommendation task. The goal is to recommend articles to authors on the arXiv. We are using Laurent's dataset of user trajectories (or sessions) through arXiv articles.

## Back-of-the-envelope comparison of the arXiv dataset to natural language modeling

In natural language modeling, a standard corpus is the Penn treebank. For timeseries models, vocabulary sizes for this dataset are $10,000$ to $30,000$ words. There are $\sim 42,000$ sentences in the training set. Sentences are approximately of length 21.

For the arXiv dataset, we use a vocabulary of $\sim 72,000$ articles, $\sim 28,000$ users, and approximately 50 articles per user. The larger vocabulary size will lead to longer training times. It is also unclear how the dependencies in the longer timeseries will affect generalization. However, the vocabulary size and length of timeseries are of comparable magnitudes so we expect our method to work.

## Validation and test method

Rather than hold out a subset of clicks, we validate on entire user timeseries of items and do next-step prediction. A timeseries is held out, and the model sees one step at a time, and generates a ranking of items to predict the next item.

## Noise is the issue: why vanilla LSTMs outperform variational autoencoders, and why this comparison is unfair

We expect vanilla LSTM networks and attention models to outperform our probabilistic variational autoencoder. This is an important issue: we emphasize that they are qualitatively very different models.

Vanilla LSTM models and attention LSTMs will outperform our models as they mix inference and model. Furthermore, our probabilistic framework adds a Kullback-Leibler regularizer to the cross-entropy loss that is used in standard sequence to sequence models. The reparametrization trick (Kingma and Welling, 2014) injects noise into the latent variables between the inference and generative network. Neural networks are notoriously bad at dealing with noise. Thus many regularization methods such as dropout (Hinton, 2014) artificially add noise to enable the networks to generalize better at test time.

While attention LSTMs may outperform our models, we argue that they suffer at test time and in terms of interpretability of latent states. Namely, visualizing the hidden states of a trained LSTM shows that the hidden states can all be clustered close togethere in latent space. This leads to issues when using models in practice: they do not generalize. For example, in the Google Inbox (inbox.google.com) auto reply sequence to sequence framework, a lot of hacks are needed to present the user with a diversity of potential replies (Corrado, 2015). This issue would not arise in variational models, as the KL regularization in the ELBO forces the model to use more of its representational capacity and push the latent variables farther apart in latent space.

### Dealing with noise: annealing, mirror autoencoders, etc.

The plan is to test various methods of training probabilistic time series models. We will try annealing methods, mirror autoencoders, and forms of regularization such as dropout.

### Incorporating side information

We will also test incorporating side information. One easy way to add this in is to concatenate the arXiv document embeddings with LDA topic vectors. This would efficiently allow us to capture content-level information in the user preferences as well.

### The plan

We have ensured that we can overfit to this dataset using a vanilla LSTM. We have trained a variational autoencoder LSTM (with latent variables at each timestep), and are exploring the fit using prior checks.

Next, we will run the vanilla word2vec model on the user trajectories to see whether our evaluation plan makes sense (visualizing latent user trajectories in embedding space).

Finally, we will add an attention mechanism to our model, and quantitatively and qualitatively evaluate it against baselines. Our goal is to submit to Recsys.

## References

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. (2015). Generating Sentences from a Continuous Space. pages 1–13.

Corrado, G. (2015). Computer, respond to this email. Google Research blog, Nov. 3, 2015.

Hinton, G. (2014). Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)*, 15:1929–1958.

Kingma, D. P. and Welling, M. (2014). Auto-Encoding Variational Bayes. *ICLR*, (Ml):1–14.