Assisting Discussion Forum Users using Deep Recurrent Neural Networks

Jacob Hagstedt P Suorra, Olof Mogren

Chalmers University of Technology, Sweden

jacob.hagstedt@gmail.com
mogren@chalmers.se

Abstract

We present a discussion forum assistant based on deep recurrent neural networks (RNNs). The assistant is trained to perform three different tasks when faced with a question from a user. Firstly, to recommend related posts. Secondly, to recommend other users that might be able to help. Thirdly, it recommends other channels in the forum where people may discuss related topics. Our recurrent forum assistant is evaluated experimentally by prediction accuracy for the end–to–end trainable parts, as well as by performing an end-user study. We conclude that the model generalizes well, and is helpful for the users.

1 Introduction

Discussion forums pose an interesting setting for human interaction. Chat systems, social media, and customer support systems are closely related, and in this paper, we will use the term "discussion forum" for all of them. These platforms play an increasingly important role for people, both in their professional and personal lives. For example, many software developers are familiar with web services such as Stack Overflow where you ask questions and other users can respond. Similar approaches are also used in customer support systems, allowing for quick turnaround time and a growing database of queries that can be made available to customers along with their responses.

In this paper, we will discuss how an automated system can help people make better use of existing platforms, and we propose a system that solves some of the associated problems. More specifically, our system helps people find their way around a discussion forum and gives intelligent suggestions on where to get the information that they need.

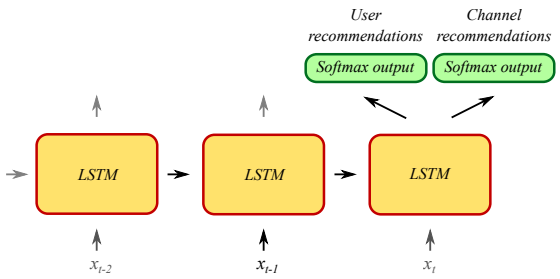

Figure 1: The layout of our recommendation model. The recommendations of users and channels are modelled as two different softmax output layers, attached to the end of a deep recurrent LSTM network modelling the input.

The proposed system is based on deep recurrent neural networks (RNNs) and solves three different problems for discussion forum users. Firstly, faced with a question from a forum user, our system can suggest related posts from other channels in the system, based on a similarity measure computed on representations learned by a Long Short Term Memory (LSTM) RNN (Schmidhuber and Hochreiter, 1997). Secondly, we train a similar network end–to–end to recommend other forum users that might be knowledgeable about the current question. Finally, the model is also trained to suggest other channels where similar discussions have been held previously.

The assistant is evaluated on data from a corporate discussion forum on the chat-platform Slack. We show experimental results by evaluating the generalization of our model, as well as performing and analysing a study based on collecting data from users who interact with the discussion forum assistant.

2 Background

A recurrent neural network (RNN) is an artificial neural network that can model a sequence of arbitrary length. The basic layout is simply a feedforward neural network with weight sharing at each position in the sequence, making it a recursive function on the hidden state h_t. The network has an input layer at each position t in the sequence, and the input x_t is combined with the the previous internal state h_{t-1}. In a language setting, it is common to model sequences of words, in which case each input x_t is the vector representation of a word. In the basic variant ("vanilla" RNN), the transition function is a linear transformation of the hidden state and the input, followed by a pointwise nonlinearity.

$$h_t = \tanh(Wx_t + Uh_{t-1} + b),$$

where W and U are weight matrices, and b is a bias term.

Basic "vanilla" RNNs have some shortcomings. One of them is that these models are unable to capture longer dependencies in the input. Another one is the vanishing gradient problem that affects many neural models when many layers get stacked after each other, making these models difficult to train (Hochreiter, 1998; Bengio et al., 1994).

The Long Short Term Memory (LSTM) (Schmidhuber and Hochreiter, 1997) was presented as a solution to these shortcomings. An LSTM is an RNN where the layer at each timestep is a cell that contains three gates controlling what parts of the internal memory will be kept (the forget gate f_t), what parts of the input that will be stored in the internal memory (the input gate i_t), as well as what will be included in the output (the output gate o_t). In essence, this means that the following expressions are evaluated at each step in the sequence, to compute the new internal memory c_t and the cell output h_t. Here "$\odot$" represents element-wise multiplication.

$$\begin{aligned}
i_t &= \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}),\\
f_t &= \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}),\\
o_t &= \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}),\\
u_t &= \tanh(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}),\\
c_t &= i_t \odot u_t + f_t \odot c_{t-1},\\
h_t &= o_t \odot \tanh(c_t).
\end{aligned} \tag{1}$$

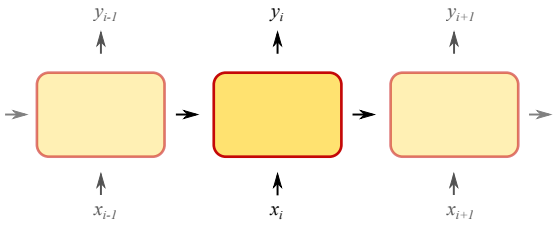

Figure 2: A recurrent neural language model. At each input x_i, the model is trained to output a prediction y_i of the next token in the sequence, x_{i+1}. In this paper, each block is a deep LSTM cell, and the network is trained using backpropagation through time (BPTT).

LSTM networks have been used successfully for language modelling (predicting the distribution of the word following after a given sequence) (see Figure 2), sentiment analysis (Tang et al., 2015), textual entailment (Rocktäschel et al., 2016), and machine translation (Sutskever et al., 2014). In the following section, we will see that the learned features are also suitable for relating forum posts to each other, and as a building block for the recommendation system in our virtual forum assistant.

3 The Recurrent Forum Assistant

In this section, we present a virtual forum assistant built using LSTM networks.

The assistant solves three different tasks in a discussion forum at an IT consultant organization. The forum is used internally and contains discussions regarding both technical topics and more everyday issues. When a user enters a question (defined simply by containing a question mark), the assistant produces one output corresponding to each task, and posts this back to the channel where the question was asked. The first task is recommending forum posts, the goal of which is to suggest related posts that might be of help to the user. The second task is to recommend other forum users that are suited to answer the question, and the third task is to suggest other forum channels where you could look for an answer to the question. See Figure 3 for an illustration of the assistant in action.

All three tasks are solved using the same underlying model, a deep recurrent LSTM network initially pretrained as a language model (see Figure 2). The pretraining is first performed using a general corpus (Wikipedia), and then using

the posts from the discussion forum. Finally the model is trained in a supervised fashion to perform the recommendation tasks (see Figure 1).

The following sections will go through how the agent solves the three different tasks.

3.1 Recommending Related Posts

The subsystem for recommending related forum posts works by first feeding each post p through the recurrent network to compute the final internal representation, $r_p = c_T$ (see Equation 1). The forum post representations are then compared using cosine similarity to get a similarity score between different forum posts:

$$sim(r_1, r_2) = \frac{r_1 \cdot r_2}{\|r_1\|\|r_2\|}. \qquad (2)$$

When posed with a question q from a user, the assistant finds the post p that maximizes $sim(q, p)$.

Representing the posts using the internal representations learned by a recurrent neural network has a number of benefits. Firstly, we can represent a sequence of arbitrary length. Secondly, the structure of the LSTM cells gives us a model that takes into account the order of the words.

3.2 End–to–End Learning of Recommendations

The second part of our virtual forum assistant is trained in an end–to–end fashion with the aim of recommending relevant **(a)** forum *users*, and **(b)** forum *channels* that might be of help to the user.

The recommendation model is built on the post recommendation model, and hence first pretrained as a language model. In order to recommend users and forum channels, we attach two multiclass classification output layers to our recurrent neural network (see Figure 1 on page 1). These are softmax layers with the number of outputs corresponding to the number of users and the number of channels in the forum, respectively. During training, the author of each post is assigned as the target value for the user recommendation layer. Similarly, the channel in which the post was made, is assigned as the target value for the channel recommendation layer. This means that we can get recommendations for forum posts, forum users, and forum channels at the same time, from the same source forum post, using the same underlying model.

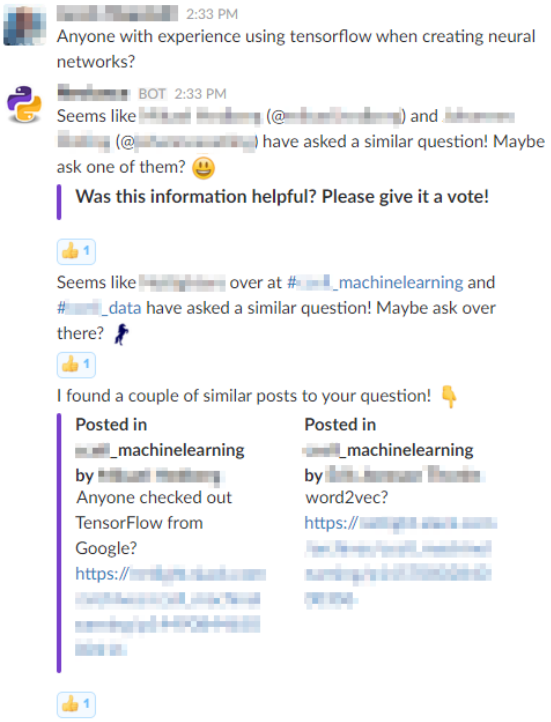

Figure 3: Screenshot of the Slack user interface when asking a question to which the recurrent assistant provides responses. Names and usernames have been anonymized.

4 Experimental Setup

This section explains the setup of the empirical study of our model. How it is designed, trained, and evaluated.

4.1 Model Layout

The same recurrent neural network is used both in the forum post recommendation step and for the recommendations for users and channels. We use a deep recurrent neural network with LSTM cells. The depth of the network is 2, and we use 650 hidden units in the LSTM cells.

For the pretraining phase, the output layer of the model is a softmax layer with 45985 outputs (the number of words in the vocabulary). For the user and channel recommendations, two softmax layers are attached to the last output of the recurrent network, one for *user* recommendations and one for *channel* recommendations (see Figure 1 on page 1). As pretraining, only the language model is trained. Then, both the recommendation output layers are trained simultaneously.

4.2 Baselines

For the related forum post recommendations, a baseline was implemented and evaluated using precomputed word embeddings from Word2Vec[1] (Mikolov et al., 2013). The precomputed model contains 300 dimensional vectors for 3 million words that were trained on the Google News corpus. For each post, a representation was computed by simply summing the vectors for each word. The forum post representations were then compared using cosine similarity (see Equation 2).

For forum user and channel recommendations, the baseline reported is a naïve solution, consistently recommending the same top-2 items; the items that maximizes the score, i.e. the 2 most common targets.

4.3 Datasets

Two datasets were used during the training; the English Wikipedia and data exported from a forum on the Slack platform.

The Wikipedia data was used to prime the model with generic English language. For this, the complete dump from 20150315 was used[2]. The dump was cleaned using Wiki-Extractor[3], and then tokenized using the Punkt tokenizer in Python NLTK.

In the discussion data from Slack, we collected all public posts made by an IT consultant organization. The discussions contain questions and answers about programming practices; different libraries and languages and what they are best suited for. The nature of the discussions are similar to that of the well known online system Stack Overflow[4], where software developers ask questions and anyone can respond. In both environments, the responses can then receive feedback and reactions.

At the time of exporting data from Slack, this forum contained 1.7 million messages written by 799 users in 664 channels. Many of these are private messages that were not used in this work. Non-public messages, inactive users (having authored less than 10 posts) and channels with fewer than 50 messages were removed, leaving 184.637 public messages, 660 users, and 321 channels that were used for training. The messages were in average 17 words long (minimum 0 and maximum 1008). A random split was made, reserving 369 posts for the validation set, and a separate export of data from the following month, resulted in 14.000 posts for the (separate) test set.

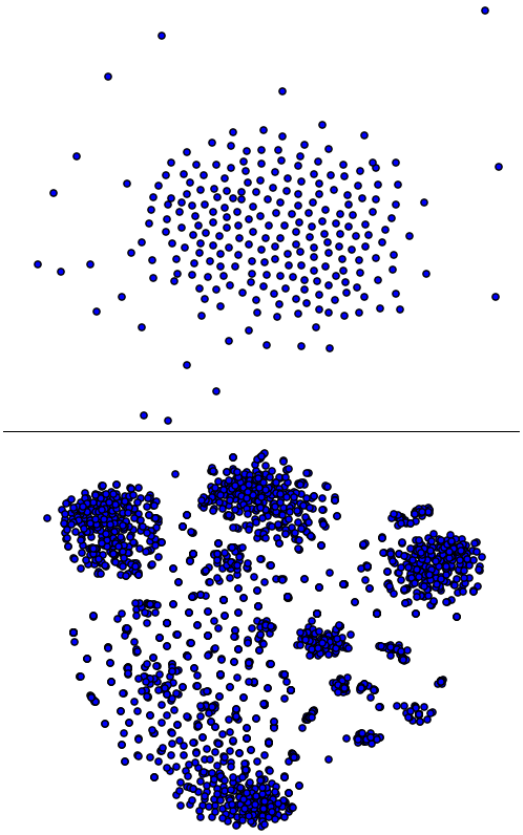

Figure 4: T-SNE projections of forum post representations.
Top: posts are represented as a sum of embeddings from Word2Vec over the words in each post.
Bottom: the internal state of an LSTM network is used as the representation.
The posts were taken from a discussion channel about mobile app development. You can see that while the word-embedding sum baseline are all clustered together, the representations created using LSTMs result in easily separable clusters.

[1] https://code.google.com/p/word2vec/
[2] https://dumps.wikimedia.org/
[3] https://github.com/bwbaugh/wikipedia-extractor
[4] https://stackoverflow.com/

Cosine	(a) Word Embedding Baseline
0.854	Having a edge on differen javascript frameworks would be very cool. We could have multiple [...]
0.848	So I have a lot of javascript that will be used across about 40 sites. [...]
0.842	Hey guys! Me myself and <user> are having a discussion regarding using Typescript with Angular.js [...]

Cosine	(b) Recurrent Forum Assistant
0.927	can someone recommend testing frameworks for Python?
0.921	Does anyone have experience in using Zend Server (for debugging) with Eclipse?
0.918	are you using any framework? such as phpspec?

Table 1: Top 3 responses from (a) the baseline method (see Section 4.2), (b) the recurrent forum assistant, when asking the question: "Do we have any experience with using angular and javascript two way databinding?". The first 15 words of each post was included. (See Section A.1 for the complete posts).

4.4 Training

Preliminary results showed that training the model on the discussion forum data alone was not enough to give good suggestions of related posts. Given the limited nature of this data, we decided to pretrain the model (as a language model) using one pass through the whole Wikipedia dump. The model was then trained for 39 epochs as a language model on the discussion data from Slack, whereafter finally the two recommendation output layers (for forum user recommendations and forum channel recommendations) were trained simultaneously for 19 epochs. Using the Wikipedia pretraining substantially improved the performance of the system. Training time was decided using early stopping (Wang et al., 1994).

Training was done with backpropagation through time (BPTT) and minibatch stochastic gradient descent.

Training the user recommendation classification was done by having the author of each forum post as the classification target. Similarly, the training target for the forum channel classification was the channel in which the corresponding post was made.

4.5 Evaluation

To evaluate the performance of the proposed virtual assistant system, two different approaches were used. Firstly, a separate test set (see Section 4.3) was constructed to evaluate the generalization of the model in the user and channel recommendations. Secondly, a user study was performed, evaluating actual performance of the agent in a live setting in the live system with users interacting with it.

When evaluating the recommendations produced by the assistant on the held–out test set, several recommendations could be reasonable choices to any one question. Therefore, we employed a top-2 testing approach, where the system was allowed to produce two recommendations for each query. If the correct target was one of the two recommendations, it was counted as "correct". The top-2 evaluation also reflects the live implementation of our system, where two recommendations are always produced.

In the user study, the agent collected a number of data-points for the evaluation after each recommendation produced. These included an identifier of the questioner, the agent's response, a timestamp, what kind of recommendation that the agent provided (posts, users, or channels), and a list of reactions that was provided by the users towards the agent's action. Positive and negative reactions were then counted and reported, as well as recommendations from the assistant that did not receive any user reactions. Along with each recommendation, the assistant encourages users to provide reactions to them (see Figure 3).

For the post recommendations in the user study, each question was served either by the LSTM state representation, or by the word embedding representation baseline, randomly picked with equal probability.

5 Results

This section presents the results of the experimental evaluation of the recurrent forum assistant.

Table 1 shows example forum post recommendation outputs from the assistant using **(a)** the word-embedding sum representations, and **(b)** the LSTM representations when posed with the example question:

"Do we have any experience with using angular and javascript two way databinding?".

We present the top-3 outputs from the word-embedding baseline method and from the recurrent forum assistant, along with the cosine similarity to the representation for the question.

For recommending forum users and channels, we report accuracy scores for the test set (see Table 3). The accuracy score is the percentage of recommendations performed on the previously unseen test-set, compared to the naïve baseline of consistently recommending the top-2 users or channels respectively; the fixed recommendation that maximizes the score.

We also report results from the user study (see Table 2). For each recommendation that the assistant post in the forum, positive and negative reactions are counted. If more than 60 minutes go without a reaction, we count this as one "No reaction". Hence, you can get more than one positive reaction and more than one negative reaction for each recommendation, but only one "No reaction".

In total, 123 reactions were collected in the user study.

	Positive	**Negative**	**No reaction**
Users	70.4%	6.1%	23.5%
Channels	80.9%	4.8%	14.3%
Posts LSTM	42.1%	47.4%	10.5%
Posts W2V	35.7%	57.1%	7.1%

Table 2: The results from the live user study. Percentage is based on the total number of reactions to the agent's actions (and an action from the agent that resulted in no reaction from users is counted as "no reaction"). For users and channels recommendations most reactions are positive, suggesting that our assistant is useful to the forum users.

	User	**Channel**
Recurrent assistant	14.39%	22.01%
Naïve baseline	2.46%	5.54%

Table 3: Accuracy of the recommendations from the agent regarding forum users and channels, respectively, on the separate test set. The proposed assistant beats the naïve baseline by a large margin.

6 Related Work

Machines that can communicate with humans in natural language have fascinated people a long time. Alan Turing defined and gave name to a test that he meant aimed to measure a machine's ability to exhibit intelligent behavior (Turing, 1950). Taking place in a chat setting, the task is for the machine to appear like a human to a panel of judges. The test has been debated by some for not measuring intelligent behavior at all. However, the topic is at the heart of artificial intelligence, and a machine that can communicate in natural language is not only fascinating, but can also be very useful.

There has been a number of different approaches to neural representations of sentences and documents. A common way of representing sequences of words is to use some form of word embeddings, and for each word in the sequence, do an element-wise addition (Mitchell and Lapata, 2010). This approach works well for many applications, such as phrase similarity and multi-document summarization (Mogren et al., 2015), even though it disregards the order of the words. Paragraph vectors (Le and Mikolov, 2014) trains a model to predict the word following a sequence. The paragraph vectors are trained, using gradient descent, at the same time as the word vectors in the model. Our approach for embedding forum posts (as described in Section 3) is more similar to (Cho et al., 2014), where the authors use a recurrent LSTM network for machine translation, by encoding an input sequence into a fixed representation which is then decoded into a sequence in another language. Other approaches have been using convolutional neural networks (Blunsom et al., 2014), and sequential denoising autoencoders (Hill et al., 2016).

Dialog systems, also known as conversational agents, typically focus on learning to produce a well-formed response, and put less emphasis on the message that they convey in their responses. Partially observed Markov descision processes (POMDPs) have been applied to this task (Young et al., 2013), but they typically require handcrafted features. (Sordoni et al., 2015) used a recurrent encoder–decoder model to perform response generation from questions as input, and training the model using two posts as input and the following response as target. (Serban et al., 2016) presented a dialog system built as a hierarchical recurrent LSTM encoder–decoder, where the dialogue is seen as a sequence of utterances, and each utterance is modelled as a sequence of words.

QA systems attempt to give the answer to a question given a knowledgebase as input. (Hermann et al., 2015) used LSTM networks with an attention mechanism to answer questions about an input text. (Bordes et al., 2015) used memory networks to answer questions with data from Freebase.

7 Discussion

The results in the empirical evaluation of the system proposed in this paper show some interesting points.

The accuracy of the model on the test set (see Table 3) shows that the model beats the naïve baseline by a large margin for forum user and channel recommendations. Since we employed a top-2 testing approach (see Section 4.5), the baseline system were allowed to recommend the two most frequent targets, resulting in a score of 2.46% and 5.54%, for user and channel recommendations, respectively. However, with the corresponding accuracy scores of 14.39% and 22.01% for the recurrent forum assistant, we have a solid improvement.

The user study (see Table 2) shows that forum users give positive reactions to most recommendations made by the recurrent assistant when recommending forum users and channels (70.4% and 80.9%, respectively). Some recommendations did not receive any reactions, and although people were encouraged to give reactions, it is hard to say what the reason is for the missing ones. However, even if you interpret each missing reaction as one negative reaction, the positive reactions are still many more.

For the related post recommendations, the number of positive user reactions are much lower (42.1% and 35.7%, respectively). We note that the two evaluated methods for representing forum posts give recommendations of comparable quality. You can see in the examples in Table 1 that using the LSTM state to represent forum posts results in a system that is able to generalize very well, which might be desirable or not depending on application. The system finds responses that are less specific compared to the ones found by using the word embedding representations. This seems like a reasonable result from a network that was trained as a language model. E.g: a language model will compute a similar distribution over the next word after observing the word "Python", as compared to observing the word "Java". In a forum post recommendation system, however, the difference between the two are crucial. Even if the network was in the end trained to recommend users and channels (something that we presumed would help learn features that were well suited also for the forum post recommendations), perhaps some other strategy for training the network, using more direct feedback from the learning objective, would work better for this task.

Figure 4 shows clustering of forum posts created with T-SNE, using (top) word-embedding representations, and (bottom) LSTM representations. The bottom plot shows how forum posts are clearly separated into clusters based on the LSTM representations, but this technique seems unable to separate the posts into clusters using word-embeddings. We believe that the reason might be connected to the observation in previous paragraph, as the LSTM representations are trained using a different objective.

In this paper, we stated the problem (and the three subproblems) as the task of finding relevant information (posts, users, and channels) whithin the current forum. The same approach can be used to find things from other sources. In the same setting, recommending posts in other forums, or pages on Wikipedia would be reasonable choices. In a customer support setting, a database of predefined statements or solution suggestions would be more suitable. With subtle changes to the implementation, the system can learn to choose from a number of output templates, and then fill in the related information from the context.

8 Conclusions

In this paper, we have proposed a virtual assistant for discussion forum users, built using deep recurrent neural networks with LSTM cells. Our solution relies heavily on learning useful representations for the data in discussion forums.

We found that using the representations from a deep recurrent neural network can be useful for the retrieval of relevant posts. However, in this particular task we found that using a representation based on summing word-embeddings works comparably well. We also found that pretraining the RNN as a language model with a general corpus such as Wikipedia gave substantially better suggestions of related posts.

Given an input question, the proposed model is able to give good recommendations for forum users and forum channels. This is evaluated both as a prediction task on an unseen test-set, and in a user study where we measure user reactions when interacting with our assistant.

Our joint model learns to produce recommendations for both users and channels, and generalize well to unseen data.

Our results from the user study clearly shows that the users find the suggestions from the assistant to be positive and useful. More experiments and A/B testing is left for future work to determine how the assistant can create the most useful suggestions.

In this work, we have taken an approach that we have not seen in previous work. Our aim was to create a useful virtual assistant for professional users of a discussion forum in an IT organization, and to help point users in the right directions for further reading. Vast amounts of knowledge can potentially reside inside a discussion platform, but the tools for navigating it are often primitive at best. We have seen that some of the tasks otherwise performed by helpful forum members can also be performed by a virtual recurrent forum assistant.

8.1 Future Work

Even though we have presented ways to learn good representations to perform recommendations of forum users and channels, more research is needed to find out how to best learn the representations for the post recommendation task.

We are currently working on a complete conversational agent that generates responses using a sequence–to–sequence learning approach with an attention mechanism. We believe that this, in combination with using external sources of information such as Wikipedia pages or databases containing information for customer support, can result in a promising virtual assistant.

Another exciting direction for this research will be to use the collected data from user reactions and create a model using deep reinforcement learning that can improve as it collects more data.

Acknowledgments

This work has been done within the project “Data-driven secure business intelligence”, grant IIS11-0089 from the Swedish Foundation for Strategic Research (SSF).

References

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166.

Phil Blunsom, Edward Grefenstette, and Nal Kalchbrenner. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075.

Kyunghyun Cho, Bart van Merrienboer, aglar Glehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *EMNLP*, pages 1724–1734. ACL.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc.

F. Hill, K. Cho, and A. Korhonen. 2016. Learning Distributed Representations of Sentences from Unlabelled Data. *ArXiv e-prints*, February.

Sepp Hochreiter. 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In Tony Jebara and Eric P. Xing, editors, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196. JMLR Workshop and Conference Proceedings.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Olof Mogren, Mikael Kågebäck, and Devdatt Dubhashi. 2015. Extractive summarization by aggregating multiple similarities. In *Recent Advances in Natural Language Processing*, page 451.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *International Conference on Learning Representations*.

Jürgen Schmidhuber and Sepp Hochreiter. 1997. Long short-term memory. *Neural computation*, 7(8):1735–1780.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In Dale Schuurmans and Michael P. Wellman, editors, *AAAI*, pages 3776–3784. AAAI Press.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. In Rada Mihalcea, Joyce Yue Chai, and Anoop Sarkar, editors, *HLT-NAACL*, pages 196–205. The Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432.

Alan M Turing. 1950. Computing machinery and intelligence. *Mind*, 59(236):433–460.

C. Wang, S. S. Venkatesh, and J. S. Judd. 1994. Optimal stopping and effective machine complexity in learning. In *Advances in Neural Information Processing Systems 6*. Morgan Kaufmann.

Stephanie Young, Milica Gasic, Blaise Thomson, and John D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

A Supplemental Material

This section provides some insight in the data that was used for this research.

A.1 Similar Posts

See table 4

A.2 Representative Posts from Clusters

This subsection contains representative posts from the different clusters that can be identified in (Fig 4). Headlines have been manually created.

A.2.1 Links

Im not really a android developer, but I found their training really useful when I started playing around with it $< weblink >$

Have anyone here used the event bus framework "otto" $< weblink >$

or the similar framework: $< weblink >$

here's a really well-explained description of MVP on Android. He uses otto and retrofit: $< weblink >$ Tip: $< weblink >$

A.2.2 Greetings and Thanks

oh great!

Hello!

Sweet!

Nice!

thanks!

sweet, onDetachedFromWindow looks to be working!

Hahaha awesome!

exactly!

good luck!

Absolutely!

good work!

A.2.3 Statements/Claims

oh, ViewPagers are a lot more complicated (we are preparing for Android 6 with the "new" permission system, so need to figure out all the places to guard :))

Normally I'd use Titanium Backup but as I said I'm a bit reluctant to rooting the phone at this point

I would say that you should just test it out! Easiest way of finding out

But yes it is stunning

so you don't have to set up the exchange account every time

thanks guys. We're expecting roughly 10-30.000 pushes per day, so not really high numbers. also delays are not so much of a problem and servers don't need to scale so fast. We have a couple of apps that need to be supported, but so far not too many pushes are expected (might change at a later point though)

I know that you can use lilbraryjars and injars for JAR files, but it doesn't seem to work for AAR files

(we only have a couple of logic tests so far though)

Yes, it's in the the support library (v7)

Yeah RecyclerView is the way to go, even if your list is pretty simple. All the new cool stuff for lists are implemented for the RecyclerView:)

that is a view animation. Try view.animate().alpha(float).setDuration(int).start() (perhaps more options)

Intet Broadcasts are crazy powerful if used correctly

I think they're using MathJax SDK in the app (3rd party math rendering tool)

RESTClient should maybe be renamed "HTTP-Client", it makes the calls, using retrofit

we are shipping native binaries for armeabi alongside armeabi-v7a but testing audio input/output in the emulator is not optimal so would love to have a device to test it on

A.2.4 Questions

Anyone with experience with proguard and amazon lambda functions? When I activate optimization my lambda functions fails with all data posts. No exceptions, but data is not comited correctly.

Cosine	(a) Word Embedding Baseline
0.854	Having a edge on differen javascript frameworks would be very cool. We could have multiple edges related to them, one per technology :simple_smile:
0.848	So I have a lot of javascript that will be used across about 40 sites. The javascript contains a lot of mandatory, minimum functionality for tracking user behaviour. But to fit a variety of additional needs that each individual site has, I want to make the tracking part of the javascript extendible-pluginable. The goal is that the sites can both add data to automated tracking provided by the javascript, and track events that are not automatically tracked by my javascript. So I was wondering if anyone has any good articles outlining a simple and-or good plugin architecture for javascript. Or even better, if anyone in $< company >$ has done something like this before?
0.842	Hey guys! Me myself and $< user >$ are having a discussion regarding using Typescript with Angular.js or not. Can you share knowledge like adv. and disadv. of using plain Javascript or Typescript! Our client uses Typescript so we want to use the same, to maximize compatibility with their way of doing web development. A good update on the two would help us in making sure we can develop fast and without delay!

Cosine	(b) Recurrent Forum Assistant
0.927	can someone recommend testing frameworks for Python?
0.921	Does anyone have experience in using Zend Server (for debugging) with Eclipse?
0.918	are you using any framework? such as phpspec?

Table 4: Top 3 responses from the recurrent assistant using (a) word-embedding sum representation (see section 4.2), and (b) the LSTM representation, when asking the question: "Do we have any experience with using angular and javascript two way databinding?". The Cosine column shows the cosine similarity to the query, using the respective representation.

$< user >$: Actually, I wonder why you are not using the ViewPager component that comes with Android?
- It automatically pre-loads 1..n fragments to ensure that they are ready-to-go when the user swiped from left or right
- It can embed a custom transformer that handles how the animation should happen
- It's super smooth and optimized to perfection, memory wise. With an adapter and a cached placeholder.. it will work like magic. We leveraged it for swiping through 1.. 100 images that were dynamically downloaded.
- It embeds necessary callbacks when cards change

$< user >$: $< user >$ Can the ViewPager be used to swipe the card(fragment) in any direction? (not just left or right).

not sure i follow, but you are wrapping a Java library in C++? id implement a jni class that implements the callback interface and where c++ objects registers as listeners

Hi, I have the following problem:
$< channel >$
Do anyone know a solution for this? I can't use WebView.HitTestResult result = webview.getHitTestResult();
because I reuse an old web view.

Does anyone have experience about travis? I have problem with dependencies for modules.

In RxJava, why would the order you place your observables when doing a merge matter?
If I do Observable.merge(A,B), it runs both in parallel as intended, but in merge(B, A) it waits for B to complete(or maybe submit item) before A starts.