Sequence to Sequence Learning for Query Expansion

Salah Zaiem  
Ecole Polytechnique  
Palaiseau, France  
mohamed-salah.zaiem@polytechnique.edu

Fatiha Sadat  
Université du Quebéc a Montreal  
201 Avenue du Président Kennedy, Montreal  
sadat.fatiha@uqam.ca

Abstract
As fas as we are aware, using Sequence to Sequence algo-  
rithms for query expansion has not been explored yet in In-  
formation Retrieval literature. We tried to fill this gap in the  
literature with a custom Query Expansion system trained and  
tested on open datasets. One specificity of our engine com-  
pared to classic ones is that it does not need the documents  
to expand the introduced query. We test our expansions on  
two different tasks : Information Retrieval and Answer pres-  
election. Our method yielded a slight improvement in perfor-  
mance in both two tasks . Our main contributions are :

• Starting from open datasets, we built a Query Expansion  
training set using sentence-embeddings-based Keyword  
Extraction.
• We assess the ability of the Sequence to Sequence neural  
networks to capture expanding relations in the words em-  
beddings’ space.

We afterwards started a quantitative and qualitative anal-  
ysis of the weights learned by our network. In the second  
part, I will discuss what is learned by a Recurrent Neu-  
rnal Network compared to what we know about human lan-  
guage learning.

Related Work
Relevance feedback has been a popular choice for query  
expansion, starting with the Rocchio Algorithm (Salton  
1971) in SMART Information Retrieval System. Using a set  
of relevant and non relevant documents, the original query  
vector is modified.

Recently, the introduction of word embeddings (Mikolov  
et al. 2013) allowed new possibilities for Query Expansion.  
The distributed representations of the words in a query  
made it possible to produce expansions without extracting  
them from the documents. Using the centroid of the words  
introduced and cosine-similar tokens, Kuzi and al (2016)  
proposed a document-independent expansion method.

Sequence to sequence architectures
Sequence to Sequence is a neural architecture very popular  
in machine translation since it has achieved state of the art  
results. Proposed by Sutskever and al (2014), it consists in  
a two-component model using recurrent neural networks to  
link variable-length input sequences to variable-length out-  
put sequences. The introduced sequence gets encoded by the  
first component into a vectorial representation. Therefore,  
the decoder transforms that vector into the target sequence.  
At each step the next token maximizes :

\[ p(y_i | y_1, ..., y_{i-1}, x) = g(y_{i-1}, s_i, c) \]

where \( s_i \) is the i-th hidden state of the encoder, c the final  
vector output by the encoder representing the entire input  
sentence and \( y_i \) the i-th generated token. \( g \) is the function  
learned by the decoder.

Our Approach
Building the training set
Datasets  
We used MultiNLI (Williams, Nangia, and Bow-  
mam 2018) and SNLI (Bowman et al. 2015). For both cor-  
pora, we naturally eliminate pairs classified as contradiction  
as they shall not provide relevant expansions.  
We also selected the duplicate pairs from the Quora ques-  
tion pairs dataset and trained our expansion model using the  
words that do not appear in the first formulation.  
Finally, MSCOCO (Lin et al. 2014) dataset consists in hu-  
mann annotated captions of over 120K images. Since they are  
describing the same image. We can assume the words ap-  
pearing in one description and not in the other are an event-  
ual expansion for the first annotation.

Keywords extraction  
We will use sentence embeddings  
to find out which words contribute most to the final vector.  
This computation is based on the hidden states of the en-  
coder. The last layer in the InferSent (Conneau et al. 2017)  
model is a Maxpooling one. The words chosen the most  
times will be our selected keywords.

The training
Preprocessing  
We extract the keywords from the target  
sequence, and then we remove the ones that appear in the  
source sequence. To get targets with similar lengths, we  
remove the pairs with a target having less than 3 tokens  
and limit them to 6. We finally get 520k pairs of sentence-  
expansion.
Training Model  We initiated the encoder and decoder weights with pre-trained word embeddings. We chose Glove 840B with 300 dimensioned vectors. To choose the hyper parameters, we relied mainly on the best practices given by Britz and al. (2017). We used a Bidirectional LSTM encoder with two layers of 500 hidden units. The decoder is a 2-layer LSTM with 500 hidden units. We used mini batches of 32 examples, and applied a 0.35 dropout probability in the LSTM stacks. We used Stochastic Gradient Descent as our optimizer and started with a learning rate of 0.001. The learning rate goes down with a decay of 0.5 after every epoch. There were 25 epochs of training for a total training time of 85 hours. The loss function is a Softmax cross entropy loss.

Evaluation
We tested our query expansion model on two different tasks: Information Retrieval and Answer preselection.

Information Retrieval
For this task, we will use the TREC Robust 2004 Dataset . It consists in a set of 250 queries and 528,155 documents. For the search component, we will use Apache Lucene search. We start with the queries, and we expand them using our QE system and then we check the quality of the results provided by Lucene Search using StandardAnalyzer and two different weighting schemes : TF-IDF and BM25.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-IDF without QE</td>
<td>0.2517</td>
</tr>
<tr>
<td>TF-IDF with QE</td>
<td>0.2581</td>
</tr>
<tr>
<td>BM25 without QE</td>
<td>0.2709</td>
</tr>
<tr>
<td>BM25 with QE</td>
<td>0.2783</td>
</tr>
</tbody>
</table>

Table 1: Information Retrieval results

Answer Preselection
We used the WikiQA Dataset (Yang, Yih, and Meek 2015). For each question, we start a search on the set of answers with a similarity computation. Accuracy is the proportion of relevant answers within the ten preselected hypothesis. Coverage is defined as the proportion of queries that had at least one appropriate answer among the ten hypotheses.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-IDF without QE</td>
<td>0.2871</td>
<td>0.7840</td>
</tr>
<tr>
<td>TF-IDF with QE</td>
<td>0.2889</td>
<td>0.7901</td>
</tr>
</tbody>
</table>

Table 2: Answer Preselection results

Qualitative analysis and future tracks
Our Query Expansion engine does improve the results in the different tasks we tested it in, but the progress is far from being impressive, and is logically not statistically significant. Here is a list of the issues limiting the reliability of our QE system and a few future tracks for improvement:

- The QE system fails to capture the semantic mechanisms behind Query Expansion, and therefore could not expand queries of unseen topics. This may not be that surprising as the task seems very complicated and nothing proofs that the actual embedding space ensures and holds this type of semantic relationships. The expansions are therefore learned through the examples, and the models fails to enrich queries on topics it did not witness before (45% of the queries are not expanded since no new word is added).
- This makes us think that although it may not be efficient for open topics, training this model on local entailments would yield great expansion results.
- We will explore the possibility of including the search in the training process. The progress on search would be a loss function updating the weights of the encoder-decoder network. After the first training, a second one ,based on the reward for the search, would refine the parameters of our network and make it more search-oriented. With more training data, we can expect an improvement in terms of precision (MAP) and accuracy.

References


