

the morning paper

a random walk through Computer Science research, by Adrian Colyer

Made delightfully fast by

≡ MENU

Building end-to-end dialogue systems using generative hierarchical neural network models

JULY 1, 2016 ~ ADRIAN COLYER

[Building end-to-end dialogue systems using generative hierarchical neural network models](#)

Serban et al. *AAAI 2016*

After reading a few of these papers on generative non-goal driven dialogue systems, I've ended up both impressed at the early results and the direction they point in, as well as somewhat underwhelmed at the potential for this technology to be used in real-world applications in the short-term. I'm sure most startups will be building goal-driven chatbots rather than non-goal driven conversationalists anyway. Perhaps blending the two will give the most natural feeling conversation in a goal-driven context in time.

There's one counterpoint to this opinion, which is [Microsoft's Xiaoice](#). In Xiaoice, Microsoft have managed to create a general conversational chatbot with the personality of a 17-year old girl and over 40 million registered users. Over tens of millions of conversations, Xiaoice averages 23-turn chats (whereas most AI personal assistants manage 1.5-2.5 turns).

From [Nicky Cappella's article](#) in The Stack:



Traditionally, users have come to expect machines to focus on task completion, but Xiaoice focuses on the conversation itself. She memorizes and tracks users emotional states, and even offers a 33-day breakup therapy course for people having relationship problems. In December, Xiaoice even got a job in media, becoming a trainee anchor and reporting the weather on live TV on the Chinese program 'Morning News'.

How does Xiaoice manage to be so good and so engaging, when the sample dialogues in the (2015 and 2016) papers we've been looking at don't seem to reach anything like that level? (Some users have taken over 10 minutes to realise they are not communicating with a program). I searched Google Scholar and Semantic Scholar and Microsoft Research looking for Xiaoice papers but couldn't find anything (if anybody knows of papers discussing how Xiaoice works, please let me know). One clue lies in the *amount of data* that Microsoft have access to for training Xiaoice: "Xiaoice... is essentially a big data project built on Microsoft's Bing search engine." And since Xiaoice is engaging with so many users in so many conversations, it (she?) just keeps on improving:



As of August 2015, 26 percent of data in Xiaoice's core chat software derives from her conversations with humans, leading Microsoft to conclude she has entered a self-learning and self-growing loop. "Xiaoice, by and large in terms of development of artificial intelligence, is already a huge milestone," said Dr. Hsiao-Wuen Hon, who leads the project at Microsoft.

Take a look at [this Xiaoice conversation](#) as reported in The New York Times, and then compare it to sample 3-turn conversations from today's paper (which by the way, still manages to create results that beat the competition under evaluation):

Reference (U_1, U_2)	MAP	Target (U_3)
U_1 : yeah , okay . U_2 : well , i guess i ' ll be going now .	i ' ll see you tomorrow .	yeah .
U_1 : oh . <continued.utterance> oh . U_2 : what ' s the matter , honey ?	i don ' t know .	oh .
U_1 : it ' s the cheapest . U_2 : then it ' s the worst kind ?	no , it ' s not .	they ' re all good , sir .
U_1 : <person> ! what are you doing ? U_2 : shut up ! c ' mon .	what are you doing here ?	what are you that crazy ?

Table 3: MAP outputs for HRED-Bidirectional bootstrapped from *SubTle* corpus. The first column shows the reference utterances, where U_1 and U_2 are respectively the first and second utterance in the test triple. The second column shows the MAP output produced by beam-search conditioned on U_1 and U_2 . The third column shows the actual third utterance in the test triple.

(Click to enlarge).

Let's turn our attention to today's paper now, which uses end-to-end training to build a dialogue system:



We believe that training such a model end-to-end to minimize a single objective function, and with minimum reliance on hand-crafted features, will yield superior performance in the long run. Furthermore, we focus on models which can be trained efficiently on large datasets and which are able to maintain state over long conversations.

For me there are two main takeaways. First, the use of a *hierarchy* of RNNs using one to model the sequence of utterances in the dialogue, and one to model the sequences of tokens in an individual turn. And secondly the value of bootstrapping the model using external data, which makes a significant difference to model performance.

The value of bootstrapping



In general, we find that the gains due to architectural choice are smaller than those obtained by bootstrapping, which can be explained by the fact that we are in a regime of relatively little training data compared to other natural language processing tasks, such as machine translation, and hence we would expect the differences to grow with more training data and longer dialogues.

Bootstrapping uses two sources of data: general word knowledge, and domain knowledge. In order to incorporate a general understanding of the meaning of words, the word embeddings are initialized using learned word embeddings from the Google News dataset (about 100 billion words) that have been trained using [Word2Vec](#).



The sheer size of the dataset ensures that the embeddings contain rich semantic information about each word.

A second level of bootstrapping is to incorporate some general domain knowledge from a large (non-dialogue) corpus covering similar topics and types of interactions. For the MovieTriples dataset that the authors intend their bot to converse over, they use the Question-Answer *SubTle* corpus which contains about 5.5M Q-A pairs constructed from movie subtitles.



We construct an artificial dialogue dataset by taking each $\{Q,A\}$ pair as a two-turn dialog $D = \{U_1 = Q, U_2 = A\}$ and use this to pretrain the model.

Bootstrapping gives significant gains in both perplexity and error-rate measures. The biggest gain is from *SubTle*, “we believe that this is because it trains all model parameters, unlike bootstrapping from Word2Vec, which only trains the word embeddings.”

Hierarchical models

Sordoni et al. used a hierarchical recurrent encoder-decoder architecture (HRED) for web query suggestions.



In the original framework, HRED predicts the next web query given the queries already submitted by the user. The history of past submitted queries is considered as a sequence at two levels: a sequence of words for each web query and a sequence of queries. HRED models this hierarchy of sequences with two RNNs: one at the word level and one at the query level. We make a similar assumption, namely, that a dialogue can be seen as a sequence of utterances which, in turn, are sequences of tokens.

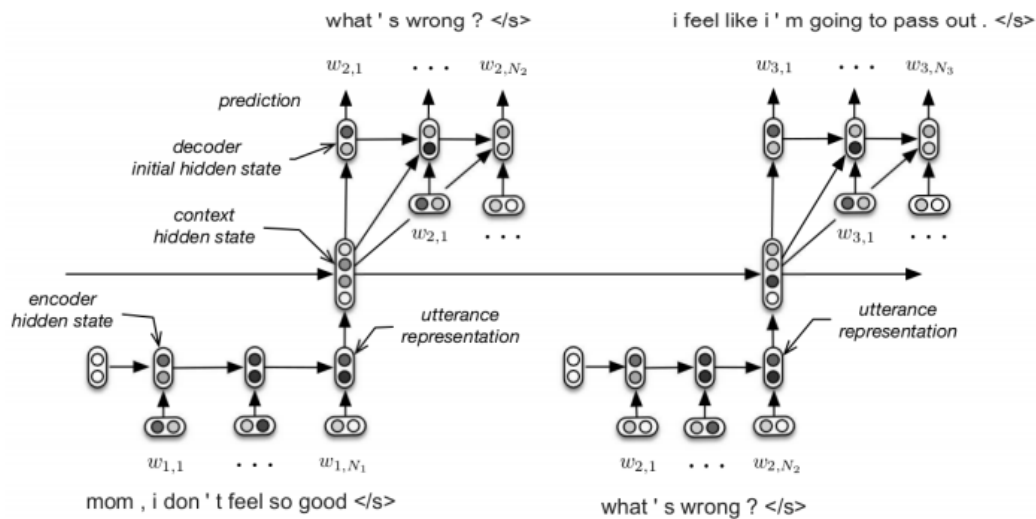


Figure 1: The computational graph of the HRED architecture for a dialogue composed of three turns. Each utterance is encoded into a dense vector and then mapped into the dialogue context, which is used to decode (generate) the tokens in the next utterance. The encoder RNN encodes the tokens appearing within the utterance, and the context RNN encodes the temporal structure of the utterances appearing so far in the dialogue, allowing information and gradients to flow over longer time spans. The decoder predicts one token at a time using a RNN. Adapted from Sordani et al. (2015a).

An *encoder* RNN maps each utterance to an utterance vector (which is just the hidden state once the last token of the utterance has been processed). A higher level *context* RNN keeps track of past utterances by iteratively processing each utterance vector. The hidden state of the context RNN therefore represents a summary of the dialogue up to the current turn. A *decoder* RNN takes the hidden state of the context RNN and produces a probability distribution over the tokens in the next utterance.



For modeling dialogues, we expect the HRED model to be superior to the standard RNN model for two reasons. First, because the context RNN allows the model to represent a form of common ground between speakers, e.g. to represent topics and concepts shared between the speakers using a distributed vector representation, which we hypothesize to be important for building an effective dialogue system (Clark and Brennan 1991). Second, because the number of computational steps between utterances is reduced. This makes the

objective function more stable w.r.t. the model parameters, and helps propagate the training signal for first-order optimization methods.

The authors use one other trick that we also saw [yesterday](#), a *bidirectional* RNN. The problem is that for longer utterances, the hidden state at the end of the utterance will not reflect as strongly tokens at the beginning of the utterance. So by running one forward chain and one backward chain (reversing the tokens) and combining the results (e.g. by concatenation) we can give equal weighting to the start and end of an utterance.

The problem of generic responses

The model has a tendency to produce generic responses such as “I don’t know” and “I’m sorry.”



There are several possible explanations for this behavior. Firstly, due to data scarcity, the models may only have learned to predict the most frequent utterances. Since the dialogues are inherently ambiguous and multi-modal, predicting them accurately would require more data than other natural language processing tasks. Secondly, the majority of tokens were punctuation marks and pronouns. Since every token is weighted equally during training, the gradient signal of the neural networks is dominated by these punctuation and pronoun tokens. This makes it hard for the neural networks to learn topic-specific embeddings and even harder to predict diverse utterances... Finally, the context of a triple may be too short. In that case, the models should benefit from longer contexts and by conditioning on other information sources, such as semantic and visual information.

This problem did *not* occur with generated stochastic samples, which contained a large variety of topic-specific words.

POSTED IN [UNCATEGORIZED](#)


[DEEP LEARNING](#)

[< PREVIOUS](#)

[Incorporating \(a\) copying mechanism in sequence to sequence learning](#)

[NEXT >](#)

[Natural language understanding \(almost\) from scratch](#)

Start a conversation ...

