Natural Language Understanding – Task 2: Dialogue

Benjamin Gallusser, Simon Buus Jensen, Frédéric Lafrance, GuanJu Li

I. Baseline model

We start by adapting an existing seq2seq implementation [1] to our needs. Concretely, this means removing the attention mechanism already included in the existing implementation, and adding a data reader for the provided Movie-DiC corpus, in addition to the already-included Cornell movie dialogue dataset [2] reader. The resulting architecture is a basic seq2seq model using 3 GRU layers of hidden dimension 1024.

Leaving the parameters from the baseline unchanged, we train on the provided Movie-DiC corpus, and perform a qualitative analysis by using the included chat mode. We find that the responses are generally of extremely poor quality. We can roughly divide the model's behavior in three phases that depend on the amount of time spent in training:

1) The *untrained phase*, which occurs at the beginning of training. The model only replies with punctuation symbols, indicating that it has not learned how to form sentences at all.
2) The *trained phase*. At this point, the model has been trained enough to have a small repertoire of about a dozen different answers. These are however very generic ("i don't know", "yes", "no", etc.).
3) The *overfit phase*. When training the model for too long, it starts to always return a single answer no matter what input is given.

We note that, surprisingly, re-enabling attention (a one-line change) has no discernible effect on the results. In any case, for the baseline, we only report results with attention enabled. The approximate number of samples used in each phase can be found in section III-B.

II. Extended model

A. Implementation

In order to combat the generic responses issue and stall the overfitting, we turn our attention towards the model described in [3]. Specifically, we use an existing implementation [4] which also has the particularity of being character-level. Such a model might help to reduce overfitting in our situation with limited training data (in total around 500,000 dialogue lines), as we are guaranteed to see most characters many times in varying contexts, while many words would occur only a few times in the entire corpus. On the other hand, we technically allow the model to generate its own words since we are operating at the level of characters. We found that this was not an issue in practice. The architecture of this model consists of 3 LSTM layers of size 1024.

B. Objective function: Maximum Mutual Information

The main contribution in [3] is the use of a different objective function, called Maximum Mutual Information (MMI). The baseline uses the standard likelihood-maximizing objective, which naturally favors more generic answers. Indeed, given the large number of responses available, the most likely ones will be those that are seen often in the corpus and that apply to many situations (questions, or prompts). By using the MMI objective function, the generative process then selects a sequence according to the following equation:

$$\hat{T} = \arg\max_{T} \left[\log P(T|S) - \lambda \log P(T)\right], \qquad \lambda \geq 0 \quad (1)$$

Where T is the decoded answer and S is the encoded question. The second term in the argmax, which is specific to the MMI objective, can be understood as a weighted penalty on generic (i.e. very probable) answers, which should hopefully help us reduce the amount of non-diverse answers such as "No.", "What?" and "I'm sorry."

C. Hyperparameters

The model has the following hyperparameters:

- *Relevance* is the weight λ on the second term of the argmax in equation 1. Setting it to 0 means using the traditional objective, while higher values penalize generic answers more and more.
- *Temperature* is a scaling parameter on predicted probabilities. A low value of this parameter (lower than 1.0) pulls apart the probabilities (making high probabilities higher and low probabilities lower), thus reducing variance and producing safer, less diverse answers. A high temperature, on the other hand, tends to produce more diverse answers.
- *Beam search width* indicates how many candidates to keep when generating the response. A value of 1 is equivalent to greedily picking the best possible word/character at each step of the generation, while higher values preserve more states at each step.

III. Evaluation

A. Metrics

We quantitatively evaluate our models on two metrics: perplexity and vector extrema distance. For the character-based model, we follow existing literature [5] and compute word-based perplexity based on the following formulas:

$$BPC = -\frac{1}{T}\sum_{t=1}^{T} \log_2 \hat{P}(c_t)$$

$$PPL_{word} = 2^{BPC \cdot \frac{N_c}{N_w}}$$

Where $\hat{P}(c_t)$ is the probability of the ground-truth character at position t as output by the model, and N_c and N_w designate

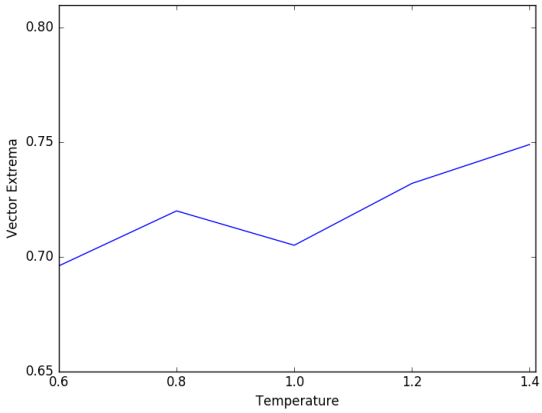

Fig. 1. Average vector extrema distance as a function of temperature

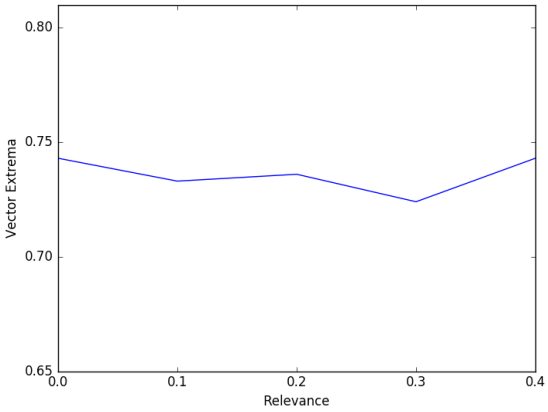

Fig. 2. Average vector extrema distance as a function of relevance

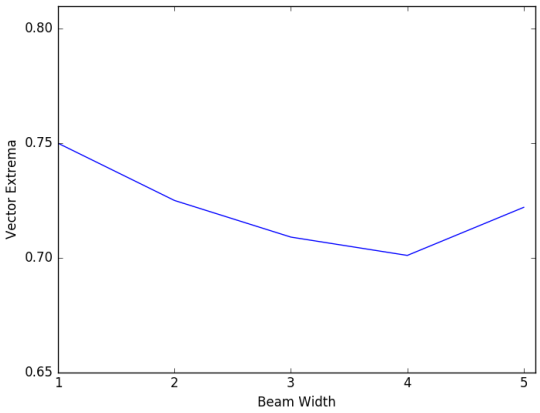

Fig. 3. Average vector extrema distance as a function of beam width

the number of characters and words in the sample, respectively. This is in effect a character-level perplexity weighted by the average length of a word.

However, we contend that perplexity is a poor measure of the quality of an open-ended dialog system such as the one we are building, since it does not capture the huge potential variability in responses. For the same reason, we do not consider overlap-based metrics such as BLEU (since those metrics are purely syntactical). We instead use the metric known as vector extrema distance [6], briefly described as follows. First, word embeddings are obtained for each word of a sentence. Then, a sentence vector is constructed by selecting, along each dimension of the embeddings, the element with the largest magnitude among all word vectors. Finally, the actual measure is computed as the cosine distance between the sentence vectors of the generated and ground-truth responses. Since this metric is embedding-based, it will not penalize answers that are on topic while being different syntactically from the ground-truth answer. Furthermore, vector extrema are intuitively superior to simple averaging for the task of constructing sentence vectors. Indeed, averaging tends to pull the representation closer to the "center" of the hyperspace, where generic words are located. On the contrary, vector extrema enhance the differences along each dimension and are thus more likely to capture the important words of the sentence.

To obtain embeddings for the purposes of this project, we use the Python package `gensim` [7], set the embedding dimensionality to 100, and train jointly on the Cornell and Movie-DiC corpora. In the case of the character-based model, we generate the response and then tokenize it to retrieve the words, whose embeddings we then look up. We consider every word that appears more than once in the joint corpus to be part of the vocabulary. Other words receive the embedding of the special "unknown" token.

B. Results and discussion

We report perplexity and vector extrema results for the baseline based on the number of batches of training (64 samples per batch) in table II. This illustrates the point we made about overfitting in section I. The drop in perplexity in the last iteration is explained by the fact that the model becomes more sure of its answers as it starts producing the same answer for every prompt. However, the vector extrema measure gets worse, indicating the general irrelevancy of the answers at this stage.

Looking at the extended model, we examine the independent effect of the various hyperparameters. Holding the others fixed to their default values (temperature 1, relevance 0, beam width 2), we vary each hyperparameter in turn. The effects on the vector extrema metric are available in figures 1, 2 and 3. We make the following observations:

- Temperature has a relatively strong effect on vector extrema measure. Somewhat counter-intuitively, lower temperature improves the metric. This could imply that we are still not properly capturing the intent that generic answers should be penalized. Alternatively, at high temperatures, we might be generating answers so diverse that they become off-topic again and are thus penalized even more heavily according to the vector extrema metric.
- Increasing beam width has a favorable effect on the vector extrema metric up to a certain point. However, when looking at the qualitative results, we notice that the

Questions	Type	Answer (baseline)	Answer (extended model)
Good morning!	General Question	what ?	you didn't say that!
Tell me something about you.	General Question	i ' m not a na .	so what do you mean?
What did you do last night?	General Question	no .	it's nothing to worry about.
Nice to meet you.	General Question	what ?	you're going to have something.
You lied to me so many times	Training Data	i ' m not a na .	no problem. you better get these.
I feel so stupid	Training Data	what ' s that ?	i'm not a woman, you haven't been with your husband. ¡person¿'s sleeping in this country.
I think we should send for a team of real scientists.	Training Data	what ?	<person>, that's what i mean. <person>?

TABLE I
A SAMPLE DIALOGUE FROM OUR BASELINE MODEL AND EXTENDED MODEL

# batches	Perplexity	Extrema
2600	201.14	0,799
5000	1689.86	0.793
8000	2737.45	**0.792**
10500	**1640.65**	0.84

TABLE II
AVERAGE METRICS ON THE BASELINE FOR VARYING AMOUNTS OF TRAINING DATA

answers become generic again. Therefore, this effect can be explained by the same reasoning as the one for the temperature hyperparameter.

Running the extended model on the best hyperparameters found above, we obtain a vector extrema distance of **0.68**. However, when manually checking the quality of the answers, we find that they are of low quality, similar to the baseline (a sample of which can be found in table I). As described above, we believe that this is due to the beam search and temperature parameters. The relevance parameter, on the other hand, helps the subjective quality of answers without compromising on the metric.

We therefore run the extended model using the default values for beam search and temperature and the value for relevance giving the best qualitative and quantitative results (0.3). A sample of the answers we obtain using these settings can be found in table I). We obtain extremely large perplexity values, causing sometimes numerical overflows. The main reason for this is that we go character-wise. Therefore, when our model is generating a word, it is generally very sure of which letter comes next. However, this character has naturally a very low chance of being the same as the character in the same position of the reference sentence. Therefore, we get heavily penalized by a perplexity measure that is not even easily comparable with the baseline since it operates on characters. **We thus choose not to report perplexity on the extended model**, and instead concentrate on the vector extrema metric. We achieve an average distance of **0.72**, a relatively good improvement over the best baseline vector extrema distance (see table II).

IV. CONCLUSION

Despite using relatively proven models, we have not been able to achieve particularly remarkable results. One problem we have run into is the relative lack of training data. In related seq2seq projects, the models are fed several million lines of dialog data during training. Our model is only fed approximately 800,000 lines of dialog data, even when adding the Cornell dataset (adding yet another similar dataset such as the IMSDB [8] would have taken us more time, as it's not readily available online). As mentioned previously, we believe that this explains the very fine line between overfitting, incoherency and good answers.

Furthermore, the search for a good metric is still on. The vector extrema metric we tried is able to distinguish between a model that produces entirely invalid output (such as the overfitted baseline) and a better model, but still has difficulties in capturing our qualitative assessments. This makes our hyperparameter search much more difficult, since we cannot rely on the value of the metric to automatically select the parameters. This lack of correlation between automated metrics and human judgments is a well-known problem in the field [9], and has not been solved yet.

APPENDIX

The following is a small guided conversation with the extended dialog model with temperature $= 1$, Beam width $= 2$ and relevance $= 0.3$, as these were the values for which our qualitative results performed best. During the guided conversation the model is simply asked questions which we find relevant provided the model's previous answer.

```
$ > Good morning!
$ why not?
$ > Not all mornings are good. Anyway,
    what did you do last night?
$ no.
$ > Ok. What did you do last night?
$ that was <person>'s.
$ > Oh, you spent last night with
  <person>?
$ that's great.
$ > I'm glad that you enjoyed it.
$ you don't understand.
$ > Why not? What happened?
$ she's the only way to leave her here.
$ > Are you in love with her?
$ <person>. <person>.
$ > Oh wow, apparently yes
```

```
$ well. <continued_utterance>. you're
  gon na kill me.
$ > Oh God, what did you do?
$ i'm going down there, <person>.
$ > Are you sure? Don't you want to
  tell me first what happened?
$ you're telling me you're a friend?
$ > Of course I'm your friend!
$ who's going to make up your neck?
$ > Hey cool down! Just tell me what
  happened!
$ but that's not my name.
$ > Ok, <person>, I'm gonna call you
  <person> again.
$ i've got something.
$ > I know man, Spit it out
$ all right. <continued_utterance>
  i'm sorry  for <person>. <person>
  and i'm gon na get this straight.
$ > I hope so. Alright I have to go
  now. See you soon
$ what is it?
$ > Never mind. Bye
$ <person>, let me give you that.
$ > Ok. What is it?
$ help me. <continued_utterance>.
  <person>! <person>?
  <continued_utterance> <person>?
$ > This is getting crazy. I'll leave
  now
$ that's not true.
$ > Look at you, shouting around here
  and almost attacking me. Of course
  you're crazy!
$ thank you.
```

REFERENCES

[1] Stanford Chatbot (GitHub)

[2] Cristian Danescu-Niculescu-Mizil and Lillian Lee, *Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs.* 2011.

[3] Li et al., *A Diversity-Promoting Objective Function for Neural Conversation Models*. 2016.

[4] Character-based Seq2seq with MMI (GitHub)

[5] Hwang and Sung, *Character-Level Language Modeling with Hierarchical Recurrent Neural Networks.* 2017.

[6] Forgues et al., *Bootstrapping Dialog Systems with Word Embeddings.* 2014.

[7] Radim Řehůřek and Petr Sojka, *Software Framework for Topic Modelling with Large Corpora.* 2010.

[8] Internet Movie Script Database

[9] Liu et al., *How NOT to evaluate your dialogue system.* 2016.