Broad Discourse Context for Language Modeling

Florian Schmidt

June 21, 2017

The purpose of this document is to agree on a general methodology for the hard word prediction task. Most likely we will always be in between two extremes: A general approach on the one hand that makes predicting hard words easier. Unless this goes at the cost of predicting non-hard words, this is a challenging task as it simply means that we improve on the decades-old task of language modeling. On the other hand, there are very specialized models focusing on a tiny aspect of the original problem that might not contribute too much to the overall perplexity of the model. We should not hesitate to look into very specific problems of the latter kind, but we should make very clear at all times which problem we are tackling and why the data we are using is an appropriate test for our models with respect to that problem.

1 LAMBADA dataset

The LAMBADA dataset will be central to this work. Still, we want to point out several biases present in the dataset:

1. A large portion of gap words are names.
2. Almost all (around 85 percent) gap words do appear in the passage before.

This is subject to the investigations Moises is currently conducting.

2 Strategy

A possible strategy could be to iterate the following steps:

1. Identify a property (of the gap words) in the LAMBADA dataset.
2. Create a new dataset in which this problem is very prominent.
3. Design a model focussing on this property, train it on the new dataset, evaluate its effectiveness on this dataset.
4. Go back to LAMBADA and test on it to see if we are doing better.
5. It would be valid to repeat the training with a mixed corpus (or objective), consisting of the new dataset and the LAMBADA training data. This should give us the best results on LAMBADA.

When coming up with alternative corpora, we are facing a challenge. We do want to stay within the book domain, which I guess means we stick to the Books dataset that LAMBADA is based on (it is only a tiny subset). That means we will always need to filter out the LAMBADA test data and ideally even all (training/dev/test) LAMBADA data. @Moises can you check whether we can get access to the original Books corpus? I only have a sentence-shuffled lower-cased version.

3 Understanding the Problem

In addition to the running experiments, it would be nice to – manually – investigate the following:

- To understand the nature of hard words "in the wild", train a model on Books and compute the per-negative-log-likelihood (effectively the per-word contributions to the perplexity). Now output some sentences with a hard word (e.g. $-\log p(w) > 20$ or similar) and investigate the sentences. The output might look like

  $$\textit{This}/8.2\ \textit{is}/4.1\ \textit{a}/3.5\ \textit{huge}/5.2\ \textit{surprise}/23.2$$

  This is essentially the data creation procedure of LAMBADA without the human in the pipeline. How many names to we get as "gaps"? How big is the bias really?

- For a reasonably trained simple baseline e.g. LSTM model, output pairs consisting of a gap word and the top-k list of predictions made by the model for that word. Do the guesses share at least the right PoS tag with the gap word? If the gap word is a name, do we predict (different) names?

  Note that these lists might look terribly even for non-hard words. Maybe do the same for some words with low perplexity as a sanity check. LMs rarely predict the right word even if perplexity is low.

4 Approaches

4.1 Continuous Cache Approach

Since we have implemented and discussed this already, let's look at this model in some detail as a start. Define an RNN so that word w_t is predicted from $\mathbf{h}_t$. The continuous cache approach [GJU16] stores pairs $(\mathbf{h}_t, w_t)$ in a cache and defines a cache word distribution as

$$p_{\text{cache}}(w|\mathbf{h}_t) \propto \sum_{i=1}^{t-1} \mathbb{I}\{w = w_i\} \exp(\theta \langle \mathbf{h}_t, \mathbf{h}_i \rangle) \tag{1}$$

which is linearly mixed into the recurrent next-word soft-max prediction distribution. The intuition is simple: If we revisit $\mathbf{h}_t$ at some time $t' > t$ (that means $\mathbf{h}_t \approx \mathbf{h}_{t'}$) , then we should probably predict w_t again.

Criticism

- They don't give any evidence why the power of their model (if any) wouldn't solely reside in the (very much not continuous) nature of the $w = w_i$. In other words, where is the comparison to a baseline with $\theta = 0$?
- Furthermore, if the hidden state similarity is meaningful, why don't they present an approach with $\exp(\theta \mathbf{h}_t^\top \boldsymbol{W} \mathbf{h}_i)$ (I am sure they tried)?
- Note that for any $(\mathbf{h}_t, w_t)$ in the cache, there is no guarantee that $P(w_t|\mathbf{h}_t)$ was particularly high. In other words, $\mathbf{h}_t$ might not be a place in which w_t is particularly likely, so why would predicting w_t when revisiting it be a good strategy? The authors hide this elegantly by introducing θ which puts strength on any last word which might be a sensible strategy (see point one).

4.2 The Topic-Modeling Direction

Our goal would be to train a topic model and a language model jointly. The work of [FWL+16] would be the prototypical example. Somehow google scholar does not list any citing papers. I don't buy that. Do [LL08] train both jointly?

In general I would suspect that the context in lambada does not have too many topic changes, ideally even none at all. However, if the topic is inferred and triggers a certain bias at the word-level distributions, this might be interesting. [QCWW16] sounds very relevant judging by the title (any citations of this?).

4.3 Latent Recurrent Models

Latent recurrent sequential models have a long history in text models (think HMMs). Training them is different from topic-models as they do not assume a document structure. While there is traditionally one latent variable per word, there are versions that have one per sentence e.g.[JHE16] [ZXS16]. Can this server as a per-sentence-topic-annotation? Current approaches include [BO14] [FSPW16] [CKD+15]

These models form a building block of the current hot topic of conversational agent as they can "bridge" the context when generating several sentences/responses. We should be interested in a similar bridging[1] functionality. [SSL+16] is seminal work in this field.

Where there are latent variables, there must be inference. Most approaches nowadays do not use EM algorithms but variational inferences with posteriors powered by neural networks. This is true for almost all approaches mentioned above. If that task is reasonable simple or the model reasonable standard, we should not hesitate to dive into this direction. It's by far the most popular one.

There are also RNN-free continuous state space models [KSS17] but I would want to see this applied somewhere else to LM before looking into it.

[1] Talking about bridging, machine translation faces a similar problem and recently this appeared: [WTWL17]

5 Proposal for Something Initial

Let's say that many gap words in LAMBADA are names and that we can easily generate a larger corpus from Books that contains many annotated names by running an NER system on it (and possibly filtering using a list of common names).

We can propose a simple name-slot-aware system that predicts words as:

$$P(w_t|\mathbf{h}_t) = \sum_{c \in \mathcal{C}} P(w_t, c|\mathbf{h}_t) = \sum_{c \in \mathcal{C}} P(w_t|c, \mathbf{h}_t) P(c|\mathbf{h}_t) \tag{2}$$

where $\mathcal{C} = \{\text{name}, \text{noname}\}$ is for now a binary classification that identifies a word as a name-slot or not. That means we can write

$$P(w_t|\mathbf{h}_t) = P(w_t|\text{name}, \mathbf{h}_t) P(\text{name}|\mathbf{h}_t) + P(w_t|\text{noname}, \mathbf{h}_t)(1 - P(\text{name}|\mathbf{h}_t)) \tag{3}$$

where $P(\text{name}|\mathbf{h}_t)$ is a binary "name-slot-predictor", $P(w_t|\text{noname}, \mathbf{h}_t)$ is our usual RNN and $P(w_t|\text{name}, \mathbf{h}_t)$ is a distribution we an design. Note that we do not need variational inference yet to predict $P(\text{name}|\mathbf{h}_t)$ as we can simply add a second supervised term to the objective

$$\mathcal{L} = \mathcal{L}^{\text{LM}}(\theta, \xi) + \lambda \mathcal{L}(\xi) \tag{4}$$

$$= \mathcal{L}^{\text{LM}}(\theta, \xi) + \lambda H[P_{\text{data}}(c), P(c|\mathbf{h}_t)] \tag{5}$$

where ξ is the parametrization of the name predictor.

Architecture A simple instantiation of this model could look like this:

- For $P(\text{name}|\mathbf{h}_t)$ simply use the hidden state with a fully connected (FC) layer and a two-class softmax.
- For $P(w_t|\text{name}, \mathbf{h}_t)$ the choice is less obvious. We can simply collect all names in the training data and use an uniform distribution. This is learning-free. Alternatively, we use a second FC+softmax-layer. This is computationally expensive but unsupervised.

  Do we need an addition term in the objective that forces $P(w_t|\text{name}, \mathbf{h}_t)$ to predict names? I hope not...

A Comment on DL Practice The usual procedure for doing this might be an attention function at the hidden-state level. That means, we define a new hidden state

$$\tilde{\mathbf{h}}_t = \gamma(\mathbf{h}_t)\mathbf{h}_t + (1 - \gamma(\mathbf{h}_t))\mathbf{h}_{\text{name}} \tag{6}$$

$$\gamma(\mathbf{h}_t) = \sigma(\boldsymbol{\xi}^\top \mathbf{h}_t + b) \in [0, 1] \tag{7}$$

where $\boldsymbol{\xi} \in \mathbb{R}^{D_h}, b \in \mathbb{R}$ are parameters and γ is essentially $P(\text{name}|\mathbf{h}_t)$ (call it deep-name-attention-mechanism :p). Words are now predicted from $P(w_t|\tilde{\mathbf{h}}_t)$ as before and we would train $\gamma(\mathbf{h}_t)$ to match the data distribution of names as above.

An important difference is that the model can shrink the impact of the name "distribution" by simply shrinking $\mathbf{h}_{\text{name}}$ in norm.

Challenge There is a practical complification with the probabilistic model above. The cross entropy loss implementations fuse the softmax and the loss because of numerical stability. This is a problem for us, as we don't want to provide logits but probabilities.

Extensions Maybe this is a blueprint for other similar models with different $\mathcal{C}$. Is there an entropy reduction when predicting words *knowing the PoS class*? Someone must have tried this. I think in general PoS helps less than one thinks but maybe it is different for hard words...

References

[BO14] Justin Bayer and Christian Osendorfer. Learning stochastic recurrent networks. *arXiv preprint arXiv:1411.7610*, 2014. arXiv.

[CKD+15] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. *CoRR*, abs/1506.02216, 2015.

[FSPW16] Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. Sequential neural models with stochastic layers. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2199–2207, 2016. NIPS.

[FWL+16] Xianghua Fu, Ting Wang, Jing Li, Chong Yu, and Wangwang Liu. Improving distributed word representation and topic model by word-topic mixture model. In *Proceedings of The 8th Asian Conference on Machine Learning*, pages 190–205, 2016.

[GJU16] Edouard Grave, Armand Joulin, and Nicolas Usunier. Improving neural language models with a continuous cache. *CoRR*, abs/1612.04426, 2016.

[JHE16] Yangfeng Ji, Gholamreza Haffari, and Jacob Eisenstein. A latent variable recurrent neural network for discourse relation language models. *CoRR*, abs/1603.01913, 2016.

[KSS17] Rahul Krishnan, Uri Shalit, and David Sontag. Structured inference networks for nonlinear state space models. 2017.

[LL08] Yang Liu and Feifan Liu. Unsupervised language model adaptation via topic modeling based on named entity hypotheses. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4921–4924, March 2008.

[QCWW16] Jipeng Qiang, Ping Chen, Tong Wang, and Xindong Wu. Topic modeling over short texts by incorporating word embeddings. *CoRR*, abs/1609.08496, 2016.

[SSL+16] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. *CoRR*, abs/1605.06069, 2016.

[WTWL17] Longyue Wang, Zhaopeng Tu, Andy Way, and Qun Liu. Exploiting cross-sentence context for neural machine translation. *CoRR*, abs/1704.04347, 2017.

[ZXS16] Biao Zhang, Deyi Xiong, and Jinsong Su. Variational neural discourse relation recognizer. *CoRR*, abs/1603.03876, 2016.