IMPROVING CTC USING STIMULATED LEARNING FOR SEQUENCE MODELING

Jahn Heymann*

Paderborn University, Germany
heymann@nt.upb.de

Khe Chai Sim, Bo Li

Google AI., USA
{khechai, boboli}@google.com

ABSTRACT
Connectionist temporal classification (CTC) is a sequence-level loss that has been successfully applied to train recurrent neural network (RNN) models for automatic speech recognition. However, one major weakness of CTC is the conditional independence assumption that makes it difficult for the model to learn label dependencies. In this paper, we propose stimulated CTC, which uses stimulated learning to help CTC models learn label dependencies implicitly by using an auxiliary RNN to generate the appropriate stimuli. This stimuli comes in the form of an additional stimulation loss term which encourages the model to learn said label dependencies. The auxiliary network is only used during training and the inference model has the same structure as a standard CTC model. The proposed stimulated CTC model achieves about 35% relative character error rate improvements on a synthetic gesture keyboard recognition task and over 30% relative word error rate improvements on the Librispeech automatic speech recognition tasks over a baseline model trained with CTC only.

Index Terms— connectionist temporal classification, stimulated learning, sequence classification

1. INTRODUCTION
Natural languages exhibit a hierarchical structure where (discrete) abstractions are composed of lower level entities which can be either discrete or continuous. For example, words are used to convey a semantic meaning and are composed of characters or symbols. These symbols can be communicated in written or oral form, where the representation is again changed and we observe a two-dimensional picture or an audio signal, i.e., a continuous representation.

Despite this hierarchical structure, nowadays recognition models usually model each level separately and then fuse the respective probabilities. For example, the acoustic model is usually not explicitly aware of the concept of words. Hybrid models assume a conditional independence given the current state and connectionist temporal classification (CTC) [1] assumes independence of each symbol given the whole audio signal. Although recurrent neural networks could theoretically resolve the label dependencies and thus implicitly learn linguistic concepts such as words, practical issues with learning long-term dependencies and limited model capacity arguably prevent them from doing so [2].

In this work, we aim to make the model aware of the context of the higher level representation by explicitly incorporating it during training. Assuming we know the alignment between the different representations, we utilize the so called stimulated learning [3, 4, 5] to constrain the state trajectory of a recurrent neural network (RNN) model to conform that of a higher level model, i.e., we stimulate the states at segment boundaries to be the same as those provided by an additional recurrent model of the higher level representation. In particular, an acoustic model is stimulated to conform the state trajectory of a language model (LM) by gradually transitioning from one LM state to another one while consuming the acoustic frames as observations. This also has the benefit that we know the state of the model at certain points in time and can train it in a segment-wise fashion—similar to truncated BPTT but without any loss of gradient information. This stimulation happens only during training time and no additional overhead during inference is introduced.

For many applications, however, we do not have such knowledge about (meaningful) segment boundaries but rather need to infer those from the observation itself. For these cases, we extend our model to full sequence training. Namely, we propose to combine it with CTC. To this extend we utilize the CTC state posteriors, i.e., the soft alignment, to obtain an attention over the state trajectory to stimulate at the right positions in time. We hypothesize that this helps the model to exploit linguistic structure and to attenuate the conditional independence assumption introduced by CTC by stimulating the model to implicitly learn label dependencies.

Our model bears many similarities to the hierarchical recurrent neural network (HRNN) [6] and even more to the hierarchical multiscale recurrent neural network (HM-RNN) [7, 8] which also tries to exploit the hierarchical structure of the data. The latter extended RNN consists of multiple layers and a latent variable which controls the current operation for each layer. Higher layers copy their state until the layer below flushes its content. In this case, the upper layer updates its state and provides the updated state as a context for the next step to the layer below. Compared to the segment-wise model we propose, the functionality and rational is very similar if the HM-RNN would learn to flush at the same segment boundaries. In fact, [8] also uses phoneme boundaries as information to guide the latent variable during training with an additional loss term. For the full sequence training, our proposed mechanism to discover boundaries is different and more explicit as it exploits prior knowledge (for example about words in the transcription).

The remainder of this paper is organized as follows. Section 2 introduces the formulation of stimulated CTC. Section 3 describes the training procedures of the proposed models. Finally, section 4 presents experimental results on a gesture keyboard recognition task and the Librispeech [9] automatic speech recognition tasks.

2. STIMULATED CTC

In a label sequence prediction task, the input feature sequence (\(X\)) and the output label sequence (\(W\)) are not always of the same length. A prediction model typically introduces a time-aligned label sequence containing repeats and blank symbols, \(Y\), so that the probability of the label sequence given the input sequence, \(P(W|X)\),

---

*Work performed while at Google.
can be decomposed into a product of conditional probabilities. For a unidirectional model, we have

\[
P(W | X) = \prod_{t=1}^{T} P(y_t | X_{1:t}, W_{1:k_t})
\]

where \(y_t\) is the time-aligned label at time \(t\), \(X_{1:t}\) is the input feature sequence up to time \(t\) and \(W_{1:k_t}\) is the corresponding output label sequence up to label \(k_t\) (the last label whose end time is less than \(t\)).

CTC is a popular sequence prediction model that has been successfully applied to automatic speech recognition [1] and keyboard gesture recognition [10]. CTC assumes that the conditional probabilities are independent of label history:

\[
P(y_t | X_{1:t}, W_{1:k_t}) \approx P(y_t | X_{1:t}) = P(y_t | h_t^{(x)})
\]

where \(h_t^{(x)}\) is the encoded input features at time \(t\) (typically using an RNN) to capture long term input history. Therefore, it has a limited capacity to learn a language model. In practice, context-dependent output units are used in combination with an external LM [11]. There are other end-to-end techniques that model the label dependencies explicitly, such as recurrent neural network transducer (RNN-T) [12, 13], listen attend spell (LAS) [14, 15] and neural transducer (NT) [16]. Specifically, RNN-T can be viewed as an extension to CTC by explicitly incorporating label dependencies:

\[
P(y_t | X_{1:t}, W_{1:k_t}) \approx P(y_t | h_t^{(x)}, h_{k_t}^{(w)})
\]

where \(h_t^{(w)}\) is the RNN encoded LM state, summarizing the label history up to label \(k_t\). However, label history has to be explicitly tracked and there is no merging of LM states due to the continuous state representation of an RNN.

In this paper, we investigate the possibility of incorporating the label history information \(h_t^{(w)}\) implicitly into \(h_t^{(x)}\) through stimulated learning [3, 4, 5] to improve a CTC model. The stimuli are generated by an auxiliary RNN, which is jointly learned with the prediction RNN. We treat \(h_t^{(w)}\) as privileged information that are only available during training and introduce an additional stimulation loss term to minimize the mean squared error (MSE) between \(h_t^{(x)}\) and \(h_t^{(w)}\). Stimulated CTC can be viewed as a kind of distillation (student-teacher) training [17], where the auxiliary RNN LM is used as a teacher to guide the student CTC RNN to learn a better underlying state representation. Like RNN-T, stimulated CTC jointly trains the RNN LM with the recognition RNN model but the LM component is only used during training.

A schematic overview of the stimulated CTC model is shown in Fig. 2.

### 3. TRAINING PROCEDURE

The stimulated CTC loss function consists of three loss terms:

\[
L_{\text{total}} = L_{\text{ctc}} + \alpha L_{\text{lm}} + \beta L_{\text{stimu}}
\]

where \(L_{\text{ctc}}\) is the standard CTC loss, \(L_{\text{lm}}\) is the LM loss and \(L_{\text{stimu}}\) is the stimulation loss. \(\alpha\) and \(\beta\) are weight factors that can be adjusted to trade-off the importance of each loss. The LM loss, which is the standard cross-entropy loss for RNN LM [18], is given by:

\[
L_{\text{lm}} = \frac{1}{K} \sum_{k=1}^{K} \log P(w_k | h_{k-1}^{(w)})
\]

where \(K\) is the label sequence length and \(\tau_k\) is the end of segment boundary for the \(k\)-th label. This allows us to perform segment-level training by splitting the observation sequence \(X_{1:T}\) into \(K\) non-overlapping segments, \(S_k\), one for each label. For the \(k\)-th segment, we train an RNN\(^1\) to predict \(w_k\) at the end of the segment and \(\text{blanks}\) elsewhere. \(h_{k-1}^{(w)}\) is used as the initial RNN state and the final RNN state after observing the segment \(S_k\) is constrained to be close to \(h_k^{(w)}\) (by using the stimulation loss in Eq. 4). If the constraint is well-satisfied, inference becomes easy as we can just treat the model as a normal RNN and fully unroll the whole sequence.

\(^1\)We use the term RNN here as a general description of a recurrent model. Any specific architecture like LSTMs, GRUs etc. can be used.
The above segment-level training decomposes a full sequence optimization problem (due to the recurrent nature of RNNs) into smaller independent optimization problems. This is similar in spirit to the work that decomposes optimization of a deep neural network into independent optimization problems, one for each layer [20, 21].

In the following, we will discuss several improvements made to the above segment-level training.

- **Stochastic Stimulation**
  It is unlikely that the recognition model will be able to transition to these states exactly since the observations themselves are assumed to be continuous. To make the model more robust against small errors in the LM state estimation, we add small Gaussian noise to the initial states during training. The variance can either be fixed upfront or generated by the auxiliary model. In the latter case, instead of MSE, negative log likelihood is used as the stimulation loss. Therefore, if the prediction of the model is inaccurate, the auxiliary model is encouraged to increase the variance. Ultimately, this should result in a variance which reflects the model uncertainty about its decision but keeps the context informative.

- **Multi-label Segments**
  Stochastic stimulation alone does not address the problem with inaccurate state trajectory estimation. When trained in a segment-wise fashion, the model never encounters a true cross segment transition since the initial state for each segment is always provided by the auxiliary model. This leads to a significant decrease in performance as we will show in the results section. To circumvent the problem, we consider training segments of multiple labels by stitching together \( m \) consecutive segments. This way, we expose the model to situations as described above during training and the model learns to cope with uncertainty in the context and also to recover from classification errors.

- **Constrained CTC loss**
  Another problem we encountered during initial experiments is that the model struggles to output the prediction at the very end of the segment. Using a per-segment CTC loss led to another problem where the model quickly learned to make the prediction at the first frame of the segment based on the LM state provided by the auxiliary model, without considering the observation. In order to prevent this behaviour, we use a constrained CTC loss that only allows labels to be emitted in the last \( 25\% \) frames of the segment, forcing the model to output blank symbols at the beginning.

### 3.2. Soft Alignment

Soft alignments are computed when computing the gradient of the CTC loss with respect to the logits. This is given by, \( \gamma_t(k) = P(y_t = w_k | X_{1:T}) \), the probability that a label \( w_k \) is aligned to time \( t \), which can be computed efficiently using the forward-backward algorithm [22]. We use this information as weight to calculate the stimulation loss as follows:

\[
\mathcal{L}_{\text{stimu}} = \frac{1}{K \cdot T} \sum_k \sum_t \gamma_t(k) \| h^{(x)}_t - h^{(w)}_k \|^2
\]

Note that for soft-alignment stimulation, we unroll the recognition network over the whole sequence. This also means that we have only one CTC loss over the whole sequence and not one per segment as in the previously described segment-level training. Overall, this is much closer to a standard RNN model, except for the stimulation of the state, which is meant to guide its trajectory.

![Fig. 2](image.png)
used to choose the optimal parameters for the learning rate, the loss weights, the context noise variance and the n-gram order of the auxiliary model if applicable. For evaluation, we decode using a prefix beam search decoder constrained by all words contained in the CMU dictionary. The metric we are interested in is the character error rate (CER). To minimize the effect of randomness, we run each experiment at least three times. We always report the best result here but none of the results had notable outliers.

The baseline model achieves a CER of 5.5% for this task (see Tbl. 2). The basic model (with fixed context noise) performs very poorly, resulting in a CER of 38.8%. However, if we provide the correct context vector also during evaluation, the CER reduces drastically to 1.9%. This supports what we suggested in Subsec. 3.1: Once the prediction is off for one segment, the model might not be able to recover due to the way we use it during inference. The proposed solution for this was to combine a certain number of segments to also expose the model to such situation already during training time. And indeed, if we only combine every two segments, the CER goes down to 6.2%. We also observe that the performance improves the more segments we combine. Notably, the model outperforms the baseline when combining four segments or more. If we also let the auxiliary model learn the parameters of the additive context noise and combine six segments, the model reaches a CER of 3.6%, i.e. an improvement of nearly 35% relative compared to the baseline. All results are shown in Tbl. 1.

Finally, we evaluate the full sequence model. The previous results already showed, that the performance improves with the number of combined segments. Tbl. 2 shows the results for fully unrolled training, i.e. when we combine all segments of a gesture. Here, no context is provided by the auxiliary model and it is only used to stimulate the state at boundaries. For a better comparison with the baseline, we also do not add noise to the context in this case. As the results show, this does not influence the performance of the stimulated model and the CER is still 3.6%. We now omit the a priori knowledge of the boundaries and use the CTC soft-alignment to stimulate the model (Subsec. 3.2) The performance is only marginally effected (3.7% CER) by this, showing the effectiveness of our proposed approach.

Table 1. CERs / % on the noisy gesture recognition dataset for different number of combined segments.

<table>
<thead>
<tr>
<th>Combined segments</th>
<th>Stimulated + fixed noise</th>
<th>Stimulated + learned noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>38.8</td>
<td>31.3</td>
</tr>
<tr>
<td>2</td>
<td>6.2</td>
<td>6.0</td>
</tr>
<tr>
<td>4</td>
<td>4.7</td>
<td>4.2</td>
</tr>
<tr>
<td>6</td>
<td>4.3</td>
<td>3.6</td>
</tr>
</tbody>
</table>

Table 2. CERs / % on the noisy gesture recognition dataset with full sequence training. Stimulated uses a priori boundary information for stimulation while stimulated CTC uses the CTC soft-alignment as attention.

<table>
<thead>
<tr>
<th></th>
<th>CTC</th>
<th>Stimulated</th>
<th>stimulated CTC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5.5</td>
<td>3.6</td>
<td>3.7</td>
</tr>
</tbody>
</table>

4.2. ASR: Librispeech

To further evaluate the full sequence model, we also compare it to a CTC baseline on the Librispeech ASR task [9]. We use all available data for training (i.e. all three training sets for a total of 900 h of speech) and evaluate on the clean and other set.

The baseline model has xx LSTM layers with xx units each and is trained using the CTC loss only. Instead of characters or phonemes, we choose subword units [24] as targets, leaving us with a total of xx classes. We compare this model to a stimulated one where we use the exact same structure but stimulate the last layer as described in Subsec. 3.2. Consequently, the auxiliary is a one layer LSTM with the same number of units and acts as an LM for the subword units. Both models are trained with Adam optimization [25] with a learning rate of 1e-5.

The results for both model decoded with different language constraints are shown in Tbl. 3. For 0-gram decoding, a zero LM weight is used to constraint the decoder to output valid words. For all cases, the stimulated model achieves much better results with gains around 30%. Both models perform at least twice as good on the clean set compared to the other set while the stimulated model loses a bit more than the non-stimulated one. Comparing the different decoding constraints, the stimulated model also does not profit as much from a stronger linguistic constrain as does the vanilla CTC model. This can be seen as an indication that the model is able to better exploit linguistic structure due to the stimulation as hypothesized.

Although our preliminary experimental results show promising improvements, we acknowledge that our results are generally worse compared to other work (e.g. [9]) and the baseline model is weak. We suspect that this is due to untuned hyperparameters in combination with the choice of subword as the output units. In addition, it may also be difficult to learn a CTC model from scratch with long utterances. Further experiments are needed to validate the effectiveness of the proposed method.

5. CONCLUSIONS

In this paper, we proposed using stimulated learning to improve a CTC model by introducing additional loss terms to encourage the model to learn implicit label dependencies. This can be viewed as a special form of student-teacher training where an RNN LM is used as a teacher to help the student CTC model learn the underlying RNN states at label emission points. Preliminary experimental results on a synthetic gesture keyboard input recognition task and the Librispeech automatic speech recognition tasks show that the proposed stimulated learning has promising potential in learning a better CTC model.
6. REFERENCES


