Towards better decoding and language model integration in sequence to sequence models

Abstract

The recently proposed Sequence-to-Sequence (seq2seq) framework advocates replacing complex data processing pipelines, such as an entire automatic speech recognition system, with a single neural network trained in an end-to-end fashion. In this contribution, we analyse an attention-based seq2seq speech recognition system that directly transcribes recordings into characters. We observe two shortcomings: overconfidence in its predictions and a tendency to produce incomplete transcriptions when language models are used. We propose practical solutions to both problems achieving competitive speaker independent word error rates on the Wall Street Journal dataset: without separate language models we reach 10.6% WER, while together with a trigram language model, we reach 6.7% WER.

Introduction

Deep learning BIBREF0 has led to many breakthroughs including speech and image recognition BIBREF1 , BIBREF2 , BIBREF3 , BIBREF4 , BIBREF5 , BIBREF6 . A subfamily of deep models, the Sequence-to-Sequence (seq2seq) neural networks have proved to be very successful on complex transduction tasks, such as machine translation BIBREF7 , BIBREF8 , BIBREF9 , speech recognition BIBREF10 , BIBREF11 , BIBREF12 , and lip-reading BIBREF13 . Seq2seq networks can typically be decomposed into modules that implement stages of a data processing pipeline: an encoding module that transforms its inputs into a hidden representation, a decoding (spelling) module which emits target sequences and an attention module that computes a soft alignment between the hidden representation and the targets. Training directly maximizes the probability of observing desired outputs conditioned on the inputs. This discriminative training mode is fundamentally different from the generative "noisy channel" formulation used to build classical state-of-the art speech recognition systems. As such, it has

benefits and limitations that are different from classical ASR systems.

Understanding and preventing limitations specific to seq2seq models is crucial for their successful development. Discriminative training allows seq2seq models to focus on the most informative features. However, it also increases the risk of overfitting to those few distinguishing characteristics. We have observed that seq2seq models often yield very sharp predictions, and only a few hypotheses need to be considered to find the most likely transcription of a given utterance. However, high confidence reduces the diversity of transcripts obtained using beam search.

During typical training the models are conditioned on ground truth transcripts and are scored on one-step ahead predictions. By itself, this training criterion does not ensure that all relevant fragments of the input utterance are transcribed. Subsequently, mistakes that are introduced during decoding may cause the model to skip some words and jump to another place in the recording. The problem of incomplete transcripts is especially apparent when external language models are used.

Model Description

Our speech recognition system, builds on the recently proposed Listen, Attend and Spell network BIBREF12 . It is an attention-based seq2seq model that is able to directly transcribe an audio recording INLINEFORM0 into a space-delimited sequence of characters INLINEFORM1 . Similarly to other seq2seq neural networks, it uses an encoder-decoder architecture composed of three parts: a listener module tasked with acoustic modeling, a speller module tasked with emitting characters and an attention module serving as the intermediary between the speller and the listener: DISPLAYFORM0

The Listener

The listener is a multilayer Bi-LSTM network that transforms a sequence of INLINEFORM0 frames of acoustic features INLINEFORM1 into a possibly shorter sequence of hidden activations INLINEFORM2 , where INLINEFORM3 is a time reduction constant BIBREF11 , BIBREF12 .

The Speller and the Attention Mechanism

The speller computes the probability of a sequence of characters conditioned on the activations of the listener. The probability is computed one character at a time, using the chain rule: DISPLAYFORM0

To emit a character the speller uses the attention mechanism to find a set of relevant activations of the listener INLINEFORM0 and summarize them into a context INLINEFORM1 . The history of previously emitted characters is encapsulated in a recurrent state INLINEFORM2 : DISPLAYFORM0

We implement the recurrent step using a single LSTM layer. The attention mechanism is sensitive to the location of frames selected during the previous step and employs the convolutional filters over the previous attention weights BIBREF10 . The output character distribution is computed using a SoftMax function.

Training Criterion

Our speech recognizer computes the probability of a character conditioned on the partially emitted transcript and the whole utterance. It can thus be trained to minimize the cross-entropy between the ground-truth characters and model predictions. The training loss over a single utterance is DISPLAYFORM0

where INLINEFORM0 denotes the target label function. In the baseline model INLINEFORM1 is the

indicator INLINEFORM2 , i.e. its value is 1 for the correct character, and 0 otherwise. When label smoothing is used, INLINEFORM3 encodes a distribution over characters.

Decoding: Beam Search

Decoding new utterances amounts to finding the character sequence INLINEFORM0 that is most probable under the distribution computed by the network: DISPLAYFORM0

Due to the recurrent formulation of the speller function, the most probable transcript cannot be found exactly using the Viterbi algorithm. Instead, approximate search methods are used. Typically, best results are obtained using beam search. The search begins with the set (beam) of hypotheses containing only the empty transcript. At every step, candidate transcripts are formed by extending hypothesis in the beam by one character. The candidates are then scored using the model, and a certain number of top-scoring candidates forms the new beam. The model indicates that a transcript is considered to be finished by emitting a special EOS (end-of-sequence) token.

Language Model Integration

The simplest solution to include a separate language model is to extend the beam search cost with a language modeling term BIBREF11 , BIBREF3 , BIBREF14 : DISPLAYFORM0

where coverage refers to a term that promotes longer transcripts described it in detail in Section SECREF16 .

We have identified two challenges in adding the language model. First, due to model overconfidence deviations from the best guess of the network drastically changed the term INLINEFORM0 , which made

balancing the terms in eq. (EQREF11) difficult. Second, incomplete transcripts were produced unless a recording coverage term was added.

Equation (EQREF11) is a heuristic involving the multiplication of a conditional and unconditional probabilities of the transcript INLINEFORM0 . We have tried to justify it by adding an intrinsic language model suppression term INLINEFORM1 that would transform INLINEFORM2 into INLINEFORM3 . We have estimated the language modeling capability of the speller INLINEFORM4 by replacing the encoded speech with a constant, separately trained, biasing vector. The per character perplexity obtained was about 6.5 and we didn't observe consistent gains from this extension of the beam search criterion.

Solutions to Seq2Seq Failure Modes

We have analysed the impact of model confidence by separating its effects on model accuracy and beam search effectiveness. We also propose a practical solution to the partial transcriptions problem, relating to the coverage of the input utterance.

Impact of Model Overconfidence

Model confidence is promoted by the the cross-entropy training criterion. For the baseline network the training loss (EQREF7) is minimized when the model concentrates all of its output distribution on the correct ground-truth character. This leads to very peaked probability distributions, effectively preventing the model from indicating sensible alternatives to a given character, such as its homophones. Moreover, overconfidence can harm learning the deeper layers of the network. The derivative of the loss backpropagated through the SoftMax function to the logit corresponding to character INLINEFORM0 equals INLINEFORM1 , which approaches 0 as the network's output becomes concentrated on the correct character. Therefore whenever the spelling RNN makes a good prediction, very little training

signal is propagated through the attention mechanism to the listener.

Model overconfidence can have two consequences. First, next-step character predictions may have low accuracy due to overfitting. Second, overconfidence may impact the ability of beam search to find good solutions and to recover from errors.

We first investigate the impact of confidence on beam search by varying the temperature of the SoftMax function. Without retraining the model, we change the character probability distribution to depend on a temperature hyperparameter INLINEFORM0 : DISPLAYFORM0

At increased temperatures the distribution over characters becomes more uniform. However, the preferences of the model are retained and the ordering of tokens from the most to least probable is preserved. Tuning the temperature therefore allows to demonstrate the impact of model confidence on beam search, without affecting the accuracy of next step predictions.

Decoding results of a baseline model on the WSJ dev93 data set are presented in Figure FIGREF13 . We haven't used a language model. At high temperatures deletion errors dominated. We didn't want to change the beam search cost and instead constrained the search to emit the EOS token only when its probability was within a narrow range from the most probable token. We compare the default setting (INLINEFORM0), with a sharper distribution (INLINEFORM1) and smoother distributions (INLINEFORM2). All strategies lead to the same greedy decoding accuracy, because temperature changes do not affect the selection of the most probable character. As temperature increases beam search finds better solutions, however care must be taken to prevent truncated transcripts.

Label Smoothing Prevents Overconfidence

A elegant solution to model overconfidence was problem proposed for the Inception image recognition architecture BIBREF15 . For the purpose of computing the training cost the ground-truth label distribution is smoothed, with some fraction of the probability mass assigned to classes other than the correct one. This in turn prevents the model from learning to concentrate all probability mass on a single token. Additionally, the model receives more training signal because the error function cannot easily saturate.

Originally uniform label smoothing scheme was proposed in which the model is trained to assign INLINEFORM0 probability mass to he correct label, and spread the INLINEFORM1 probability mass uniformly over all classes BIBREF15 . Better results can be obtained with unigram smoothing which distributes the remaining probability mass proportionally to the marginal probability of classes BIBREF16 . In this contribution we propose a neighborhood smoothing scheme that uses the temporal structure of the transcripts: the remaining INLINEFORM2 probability mass is assigned to tokens neighboring in the transcript. Intuitively, this smoothing scheme helps the model to recover from beam search errors: the network is more likely to make mistakes that simply skip a character of the transcript.

We have repeated the analysis of SoftMax temperature on beam search accuracy on a network trained with neighborhood smoothing in Figure FIGREF13 . We can observe two effects. First, the model is regularized and greedy decoding leads to nearly 3 percentage smaller error rate. Second, the entropy of network predictions is higher, allowing beam search to discover good solutions without the need for temperature control. Moreover, the since model is trained and evaluated with INLINEFORM0 we didn't have to control the emission of EOS token.

Solutions to Partial Transcripts Problem

When a language model is used wide beam searches often yield incomplete transcripts. With narrow beams, the problem is less visible due to implicit hypothesis pruning. We illustrate a failed decoding in

Table TABREF17 . The ground truth (first row) is the least probable transcript according both to the network and the language model. A width 100 beam search with a trigram language model finds the second transcript, which misses the beginning of the utterance. The last rows demonstrate severely incomplete transcriptions that may be discovered when decoding is performed with even wider beam sizes.

We compare three strategies designed to prevent incomplete transcripts. The first strategy doesn't change the beam search criterion, but forbids emitting the EOS token unless its probability is within a set range of that of the most probable token. This strategy prevents truncations, but is inefficient against omissions in the middle of the transcript, such as the failure shown in Table TABREF17 . Alternatively, beam search criterion can be extended to promote long transcripts. A term depending on the transcript length was proposed for both CTC BIBREF3 and seq2seq BIBREF11 networks, but its usage was reported to be difficult because beam search was looping over parts of the recording and additional constraints were needed BIBREF11 . To prevent looping we propose to use a coverage term that counts the number of frames that have received a cumulative attention greater than INLINEFORM0 : DISPLAYFORM0

The coverage criterion prevents looping over the utterance because once the cumulative attention bypasses the threshold INLINEFORM0 a frame is counted as selected and subsequent selections of this frame do not reduce the decoding cost. In our implementation, the coverage is recomputed at each beam search iteration using all attention weights produced up to this step.

In Figure FIGREF19 we compare the effects of the three methods when decoding a network that uses label smoothing and a trigram language model. Unlike BIBREF11 we didn't experience looping when beam search promoted transcript length. We hypothesize that label smoothing increases the cost of correct character emissions which helps balancing all terms used by beam search. We observe that at

large beam widths constraining EOS emissions is not sufficient. In contrast, both promoting coverage and transcript length yield improvements with increasing beams. However, simply maximizing transcript length yields more word insertion errors and achieves an overall worse WER.

Experiments

We conducted all experiments on the Wall Street Journal dataset, training on si284, validating on dev93 and evaluating on eval92 set. The models were trained on 80-dimensional mel-scale filterbanks extracted every 10ms form 25ms windows, extended with their temporal first and second order differences and per-speaker mean and variance normalization. Our character set consisted of lowercase letters, the space, the apostrophe, a noise marker, and start- and end- of sequence tokens. For comparison with previously published results, experiments involving language models used an extended-vocabulary trigram language model built by the Kaldi WSJ s5 recipe BIBREF17 . We have use the FST framework to compose the language model with a "spelling lexicon" BIBREF5 , BIBREF11 , BIBREF18 . All models were implemented using the Tensorflow framework BIBREF19 .

Our base configuration implemented the Listener using 4 bidirectional LSTM layers of 256 units per direction (512 total), interleaved with 3 time-pooling layers which resulted in an 8-fold reduction of the input sequence length, approximately equating the length of hidden activations to the number of characters in the transcript. The Speller was a single LSTM layer with 256 units. Input characters were embedded into 30 dimensions. The attention MLP used 128 hidden units, previous attention weights were accessed using 3 convolutional filters spanning 100 frames. LSTM weights were initialized uniformly over the range INLINEFORM0 . Networks were trained using 8 asynchronous replica workers each employing the ADAM algorithm BIBREF20 with default parameters and the learning rate set initially to INLINEFORM1 , then reduced to INLINEFORM2 and INLINEFORM3 after 400k and 500k training steps, respectively. Static Gaussian weight noise with standard deviation 0.075 was applied to all weight

matrices after 20000 training steps. We have also used a small weight decay of INLINEFORM4 .

We have compared two label smoothing methods: unigram smoothing BIBREF16 with the probability of the correct label set to INLINEFORM0 and neighborhood smoothing with the probability of correct token set to INLINEFORM1 and the remaining probability mass distributed symmetrically over neighbors at distance INLINEFORM2 and INLINEFORM3 with a INLINEFORM4 ratio. We have tuned the smoothing parameters with a small grid search and have found that good results can be obtained for a broad range of settings.

We have gathered results obtained without language models in Table TABREF20 . We have used a beam size of 10 and no mechanism to promote longer sequences. We report averages of two runs taken at the epoch with the lowest validation WER. Label smoothing brings a large error rate reduction, nearly matching the performance achieved with very deep and sophisticated encoders BIBREF21 .

Table TABREF21 gathers results that use the extended trigram language model. We report averages of two runs. For each run we have tuned beam search parameters on the validation set and applied them on the test set. A typical setup used beam width 200, language model weight INLINEFORM0 , coverage weight INLINEFORM1 and coverage threshold INLINEFORM2 . Our best result surpasses CTC-based networks BIBREF5 and matches the results of a DNN-HMM and CTC ensemble BIBREF22 .

Related Work

Label smoothing was proposed as an efficient regularizer for the Inception architecture BIBREF15 . Several improved smoothing schemes were proposed, including sampling erroneous labels instead of using a fixed distribution BIBREF24 , using the marginal label probabilities BIBREF16 , or using early errors of the model BIBREF25 . Smoothing techniques increase the entropy of a model's predictions, a

technique that was used to promote exploration in reinforcement learning BIBREF26 , BIBREF27 , BIBREF28 . Label smoothing prevents saturating the SoftMax nonlinearity and results in better gradient flow to lower layers of the network BIBREF15 . A similar concept, in which training targets were set slightly below the range of the output nonlinearity was proposed in BIBREF29 .

Our seq2seq networks are locally normalized, i.e. the speller produces a probability distribution at every step. Alternatively normalization can be performed globally on whole transcripts. In discriminative training of classical ASR systems normalization is performed over lattices BIBREF30 . In the case of recurrent networks lattices are replaced by beam search results. Global normalization has yielded important benefits on many NLP tasks including parsing and translation BIBREF31 , BIBREF32 . Global normalization is expensive, because each training step requires running beam search inference. It remains to be established whether globally normalized models can be approximated by cheaper to train locally normalized models with proper regularization such as label smoothing.

Using source coverage vectors has been investigated in neural machine translation models. Past attentions vectors were used as auxiliary inputs in the emitting RNN either directly BIBREF33 , or as cumulative coverage information BIBREF34 . Coverage embeddings vectors associated with source words end modified during training were proposed in BIBREF35 . Our solution that employs a coverage penalty at decode time only is most similar to the one used by the Google Translation system BIBREF9 .

Conclusions

We have demonstrated that with efficient regularization and careful decoding the sequence-to-sequence approach to speech recognition can be competitive with other non-HMM techniques, such as CTC.