On the Inductive Bias of Word-Character-Level Multi-Task Learning for Speech Recognition

Jan Kremer
Corti, Copenhagen, Denmark
jk@corti.ai

Lasse Borgholt
Corti, Copenhagen, Denmark
lb@corti.ai

Lars Maaløe
Corti, Copenhagen, Denmark
lm@corti.ai

Abstract

End-to-end automatic speech recognition (ASR) commonly transcribes audio signals into sequences of characters while its performance is evaluated by measuring the word-error rate (WER). This suggests that predicting sequences of words directly may be helpful instead. However, training with word-level supervision can be more difficult due to the sparsity of examples per label class. In this paper we analyze an end-to-end ASR model that combines a word-and-character representation in a multi-task learning (MTL) framework. We show that it improves on the WER and study how the word-level model can benefit from character-level supervision by analyzing the learned inductive preference bias of each model component empirically. We find that by adding character-level supervision, the MTL model interpolates between recognizing more frequent words (preferred by the word-level model) and shorter words (preferred by the character-level model).

1 Introduction

End-to-end automatic speech recognition (ASR) allows for learning a direct mapping from audio signals to character outputs. Usually, a language model re-scores the predicted transcripts during inference to correct spelling mistakes [17]. If we map the audio input directly to words, we can use a simpler decoding mechanism and reduce the prediction time. Unfortunately, word-level models can only be trained on known words. Out-of-vocabulary (OOV) words have to be mapped to an unknown token. Furthermore, decomposing transcripts into sequences of words decreases the available number of examples per label class. These shortcomings make it difficult to train on the word-level [3].

Recent works have shown that multi-task learning (MTL) [9] on the word- and character-level can improve the word-error rate (WER) of common end-to-end speech recognition architectures [3, 4, 19, 22, 23, 25, 30]. MTL can be interpreted as learning an inductive bias with favorable generalization properties [7]. In this work we aim at characterizing the nature of this inductive bias in word-character-level MTL models by analyzing the distribution of words that they recognize. Thereby, we seek to shed light on the learning process and possibly inform the design of better models. We will focus on connectionist temporal classification (CTC) [16]. However, the analysis can also prove beneficial to other modeling paradigms, such as RNN Transducers [15] or Encoder-Decoder models, e.g., [6, 10].

Contributions. We show that, contrary to earlier negative results [3, 28], it is in fact possible to train a word-level model from scratch on a relatively small dataset and that its performance can be further improved by adding character-level supervision. Through an empirical analysis we show that the resulting MTL model combines the preference biases of word- and character-level models. We hypothesize that this can partially explain why word-character MTL improves on only using a single decomposition, such as phonemes, characters or words.

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

2 Related work

Several works have explored using words instead of characters or phonemes as outputs of the end-to-end ASR model [3, 28]. Soltau et al. [28] found that in order to solve the problem of observing only few labels per word, they needed to use a large dataset of $120,000$ hours to train a word-level model directly. Accordingly, Audhkhasi et al. [3] reported difficulty to train a model on words from scratch and instead fine-tuned a pre-trained character-level model after replacing the last dense layer with a word embedding.

MTL enables a straightforward joint training procedure to integrate transcript information on multiple levels of granularity. Treating word- and character-level transcription as two distinct tasks allows for combining their losses in a parallel [22, 23, 29, 30] or hierarchical structure [14, 21, 25]. Augmenting the commonly-used CTC loss with an attention mechanism can help with aligning the predictions on both character- and word-level [4, 13, 23]. All these MTL methods improve a standard CTC baseline.

Finding the right granularity of the word decomposition is in itself a difficult problem. While Li et al. [23] used different fixed decompositions of words, sub-words and characters, it is also possible to optimize over alignments and decompositions jointly [24]. Orthogonal to these works different authors have explored how to minimize WER directly by computing approximate gradients [26, 33].

When and why does MTL work? Earlier theoretical work argued that the auxiliary task provides a favorable inductive bias to the main task [7]. Within natural language processing on text several works verified empirically that this inductive bias is favorable if there is a certain notion of relatedness between the tasks [5, 8, 27]. Here, we investigate how to characterize the inductive bias learned via MTL for speech recognition.

3 Combining word- and character-level ASR

The CTC loss is defined as follows [16]:

$$\mathcal{L}(\mathbf{x}, \mathbf{z}) := -\log \sum_{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{z})} p(\boldsymbol{\pi}|\mathbf{x}) = -\log \sum_{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{z})} \prod_t p(\pi_t|\mathbf{x}) \ , \tag{1}$$

where $\mathbf{x}$ is the audio input, commonly a spectrogram, and $\boldsymbol{\pi}$ is a path that corresponds to the ground-truth transcript $\mathbf{z}$. The squashing function $\mathcal{B}$ maps a path $\boldsymbol{\pi}$ to the output $\mathbf{z}$ by first merging repetitions and then deleting so-called blank tokens. The gradient of the CTC loss can be computed efficiently using a modified forward-backward algorithm.

Typically, π_t is a categorical random variable over the corresponding output alphabet $\mathcal{A} = \{a, b, c, ..., \epsilon\}$. Here, ϵ is the blank token which encodes the empty string. This output representation enables the model to be able to transcribe any word possible without a specified alignment. Character-level CTC models are often supplemented by an external language model that can significantly improve the accuracy of the ASR. This is because these models still make spelling mistakes despite being trained on large amounts of data [1].

By using an alphabet of words one can ensure that there are no misspellings. The alphabet could contain, for example, the most common words found in the training set. This has the advantage that any word is guaranteed to be spelled correctly and that costly re-scoring on a character-level is avoided. However, by using a word-level decoding, we can no longer predict rare or new words. In this case the model has to be content with outputting an unknown token. Another challenge when using a word-level model is label sparsity. While we will observe many examples of a single character, there will be fewer for a single word, making overfitting more likely. We aim at counter-acting these shortcomings by making use of character-level information during training, similar to Audhkhasi et al. [3].

In this work we combine word- and character-level models via an MTL loss and denote this a word-character-level model. We treat each output-level prediction as a separate task and form a linear combination of the losses. The MTL loss is then defined as

$$\mathcal{L}_{\text{MTL}}(\mathbf{x}, \mathbf{z}) := \mathcal{L}_{\text{word}}(\mathbf{x}, \mathbf{z}) + \lambda \mathcal{L}_{\text{char}}(\mathbf{x}, \mathbf{z}) \ , \tag{2}$$

where $\lambda \geq 0$ defines a hyperparameter to weight the influence of the character-level CTC loss $\mathcal{L}_{\text{char}}$ against the word-level CTC loss $\mathcal{L}_{\text{word}}$. In our experiments we set it to 1, giving equal contribution

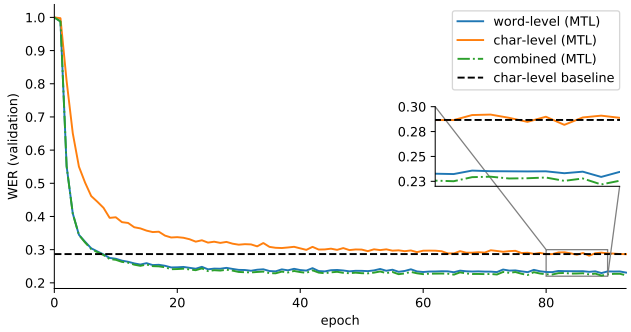

Figure 1: Multi-task learning with equal weighting on word- and character-level.

to both loss terms, but other choices may improve the performance. Alternatively, one could try to estimate this weight based on the uncertainty [18] or gradient norm [11] of each loss term. We experimented with these approaches, but did not observe any significant improvement in performance over the equally-weighted loss.

4 Experiments

We trained our models using a convolutional architecture which is based on Wav2Letter [12]. Details can be found in the appendix. Compared to recurrent neural networks, convolutional neural networks avoid iterative computation over time and suffer less from the vanishing/exploding gradient problem. They achieve comparable performance in terms of WER [12, 32]. We performed all experiments on read news articles from the Wall Street Journal (WSJ) [31]. This dataset has relatively little background noise and allows us to focus on the influence of word frequency and word length. We used the *si284* subset for training, and *dev92* for validation. For the character-level model we used 32 different characters which include the space-character and a blank token. To define the output alphabet for the word-level model, we included all words that appeared at least 5 times in the training set in addition to a blank and an unknown token. This corresponds to an alphabet of 9411 units with an OOV rate of $9\,\%$ on the training set, and $10\,\%$ on the validation set, which represents a lower bound for the achievable WER of a word-level model. For the MTL model we let word- and character-level model share every layer but the last.

To decode the output on the character- and word-level, we used greedy decoding. In order to get rid of unknown tokens in our prediction, we employed the following heuristic [22]: For each unknown token predicted on the word-level, we substituted the corresponding word on the character-level that was defined at the same time step. To compare our results we also trained word- and character-only models. For optimization we used the Adam-optimizer [20] with a learning rate of $5\mathrm{e}{-4}$ and a batch size of 16 to fit the whole model into the memory of one GPU. We applied batch normalization and dropout. For the input data, we transformed each utterance into spectrograms over $20\,\mathrm{ms}$ with a shift of $10\,\mathrm{ms}$ using 40 log-mel coefficients, standardized per spectrogram. We ran each experiment for 100 epochs, corresponding to $233,838$ updates.

MTL performance. The results of our experiments can be found in Figure 1. It shows the learning curve for the word- and character-level components by measuring the WER on the validation set. The dashed line shows the achieved WER using a character-level model without joint word-level training. We observe that MTL converges faster and to a lower WER of $23\,\%$, which is 5 percentage points lower than the character-level component of the MTL network, or the single-task character-level baseline. Using a beam search decoder with a lexicon constraint on the character-level model reduces the WER from $28\,\%$ to a WER of $24\,\%$, which is still higher than our MTL error. This shows that MTL performs favorably even without a language model. A word-level-only model achieved the same performance as the character-level baseline on this dataset. Contrary to the findings of Audhkhasi et al. [3], this shows that it is indeed possible to train a word-level model from scratch, even without a large amount of training data. While the combined decoding only gives an improvement of 0.7 percentage points in terms of WER, it eliminates unknown-token predictions which might make transcripts more readable.

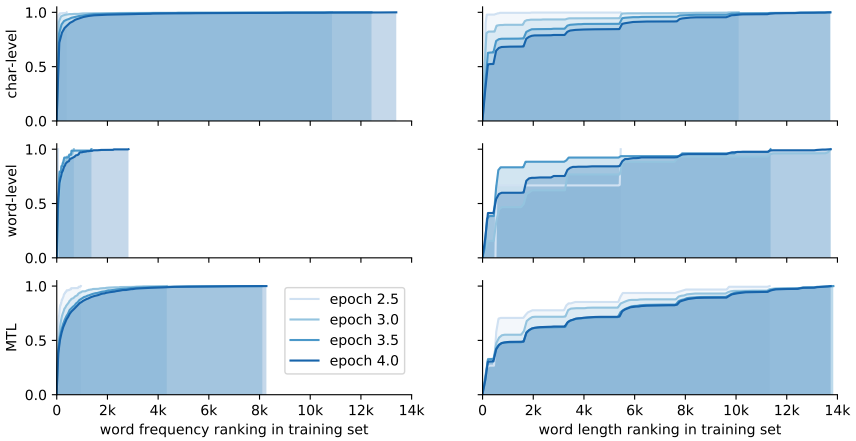

Figure 2: Comparing the CDFs of recognized words between character-, word- and MTL-model. (**left**) The distribution of recognized words during the first epochs as a function of word frequency rank. The most frequent words in the training set are on the left. (**right**) The distribution of recognized words during the first epochs as a function of word length rank. Longer words are on the right.

Characterizing the inductive bias. Arpit et al. [2] have shown that a neural network trained with stochastic gradient descent learns easier examples first. We argue that we can characterize the preference bias of our model and learning algorithm by showing which examples are easy to classify in the particular representation that each of the models is learning. Since ASR models are usually evaluated in terms of WER, we consider which words each model is learning. To this end we chose a relatively clean dataset and considered the attributes frequency and length to describe a word.

We trained each model for 4 epochs and recorded the distribution of the recognized words during training. Since we are not given a perfect alignment between speech and ground-truth transcript, we define a word as being recognized if it is both present in the greedy prediction on the validation set and the corresponding ground-truth transcript. Figure 2 shows how the distribution of recognized words changes during training. We see that the word-level model is biased towards recognizing the most common words and slowly learns less frequent words over time. This makes sense since more weight is given to the corresponding examples. While the same effect is present in the character-level model, it covers the complete support of the word frequency distribution in the same number of steps.

On the other hand for the length distribution, we see that the word-level model covers all words independent of its length within the beginning of training. The character-level model focuses strongly on shorter words before it covers the whole range of the word length distribution. If we compare the learning dynamics of both models, we find that each model learns words with different characteristics more easily. If we take a look at the MTL model, we see that it combines both biases and arrives at learning a distribution that is much more uniform across both word frequency and word length. We hypothesize that putting more emphasis on the tail of each of these distributions combines the strengths of the two models and makes them perform better, especially in distributions that follow a power law such as word frequency rank.

5 Conclusion

In contrast to earlier studies in the literature, we found that, even on a relatively small dataset, training on a word-level can be feasible. Furthermore, we found that combining a word-level model with character-level supervision in MTL can improve results noticeably. To gain a better understanding of this, we characterized the inductive bias of word-character MTL in ASR by comparing the distributions of recognized words at the beginning of training. We found that adding character-level supervision to a word-level interpolates between recognizing more frequent words (preferred by the word-level model) and shorter words (preferred by the character-level model). This effect could be even more pronounced on harder datasets than WSJ, such as medical communication data where many long words are infrequent, but very important. Further analysis of word distributions in terms of pitch, noise and acoustic variability could provide additional insight.

References

[1] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos, E. Elsen, J. Engel, L. Fan, C. Fougner, A. Y. Hannun, B. Jun, T. Han, P. LeGresley, X. Li, L. Lin, S. Narang, A. Y. Ng, S. Ozair, R. Prenger, S. Qian, J. Raiman, S. Satheesh, D. Seetapun, S. Sengupta, C. Wang, Y. Wang, Z. Wang, B. Xiao, Y. Xie, D. Yogatama, J. Zhan, and Z. Zhu. Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin. In *ICML*, 2016.

[2] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien. A Closer Look at Memorization in Deep Networks. In *ICML*, 2017.

[3] K. Audhkhasi, B. Ramabhadran, G. Saon, M. Picheny, and D. Nahamoo. Direct Acoustics-to-Word Models for English Conversational Speech Recognition. In *Interspeech*, 2017.

[4] K. Audhkhasi, B. Kingsbury, B. Ramabhadran, G. Saon, and M. Picheny. Building Competitive Direct Acoustics-to-Word Models for English Conversational Speech Recognition. In *ICASSP*, 2018.

[5] I. Augenstein, S. Ruder, and A. Søgaard. Multi-Task Learning of Pairwise Sequence Classification Tasks over Disparate Label Spaces. In *NAACL*, 2018.

[6] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio. End-to-end attention-based large vocabulary speech recognition. In *ICASSP*, 2016.

[7] J. Baxter. A Model of Inductive Bias Learning. *JAIR*, 2000.

[8] J. Bingel and A. Søgaard. Identifying beneficial task relations for multi-task learning in deep neural networks. In *EACL*, 2017.

[9] R. Caruana. Multitask Learning. *Machine Learning*, 1997.

[10] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *ICASSP*, 2016.

[11] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich. GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks. In *ICML*, 2018.

[12] R. Collobert, C. Puhrsch, and G. Synnaeve. Wav2Letter: An End-to-End ConvNet-based Speech Recognition System. *arXiv:1609.03193 [cs.LG]*, 2016.

[13] A. Das, J. Li, R. Zhao, and Y. Gong. Advancing Connectionist Temporal Classification with Attention Modeling. In *ICASSP*, 2018.

[14] S. Fernández, A. Graves, and J. Schmidhuber. Sequence Labelling in Structured Domains with Hierarchical Recurrent Neural Networks. In *IJCAI*, 2007.

[15] A. Graves. Sequence Transduction with Recurrent Neural Networks. *arXiv:1211.3711 [cs.NE]*, 2012.

[16] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *ICML*, 2006.

[17] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng. Deep Speech: Scaling up end-to-end speech recognition. *arXiv:1412.5567 [cs.CL]*, 2014.

[18] A. Kendall, Y. Gal, and R. Cipolla. Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. In *CVPR*, 2017.

[19] S. Kim, T. Hori, and S. Watanabe. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *ICASSP*, 2017.

[20] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015.

[21] K. Krishna, S. Toshniwal, and K. Livescu. Hierarchical Multitask Learning for CTC-based Speech Recognition. *arXiv:1807.06234 [cs.CL]*, 2018.

[22] J. Li, G. Ye, R. Zhao, J. Droppo, and Y. Gong. Acoustic-to-word model without OOV. In *ASRU*, 2017.

[23] J. Li, G. Ye, A. Das, R. Zhao, and Y. Gong. Advancing Acoustic-to-Word CTC Model. In *ICASSP*, 2018.

[24] H. Liu, Z. Zhu, X. Li, and S. Satheesh. Gram-CTC: Automatic Unit Selection and Target Decomposition for Sequence Labelling. In *ICML*, 2017.

[25] R. Sanabria and F. Metze. Hierarchical Multi Task Learning With CTC. *arXiv:1807.07104 [cs.CL]*, 2018.

[26] M. Shannon. Optimizing Expected Word Error Rate via Sampling for Speech Recognition. In *Interspeech*, 2017.

[27] A. Søgaard and Y. Goldberg. Deep multi-task learning with low level tasks supervised at lower layers. In *ACL*, 2016.

[28] H. Soltau, H. Liao, and H. Sak. Neural Speech Recognizer: Acoustic-to-Word LSTM Model for Large Vocabulary Speech Recognition. In *Interspeech*, 2017.

[29] S. Toshniwal, H. Tang, L. Lu, and K. Livescu. Multitask Learning with Low-Level Auxiliary Tasks for Encoder-Decoder Based Speech Recognition. In *Interspeech*, 2017.

[30] S. Ueno, H. Inaguma, M. Mimura, and T. Kawahara. Acoustic-to-Word Attention-Based Model Complemented with Character-Level CTC-Based Model. In *ICASSP*, 2018.

[31] P. C. Woodland, J. J. Odell, V. Valtchev, and S. J. Young. Large vocabulary continuous speech recognition using HTK. In *ICASSP*, 1994.

[32] Y. Zhang, M. Pezeshki, P. Brakel, S. Zhang, C. Laurent, Y. Bengio, and A. C. Courville. Towards End-to-End Speech Recognition with Deep Convolutional Neural Networks. In *Interspeech*, 2016.

[33] Y. Zhou, C. Xiong, and R. Socher. Improving End-to-End Speech Recognition with Policy Learning. In *ICASSP*, 2018.

A Architecture

The architecture we used throughout the paper is based on Wav2Letter [12]. It is shown in Table 1. Different from the original setup we use 2D convolutions in the first layers following the input and a slightly larger network. We apply dropout, batch normalization and ReLU activations to every layer but the input and last layer. We use a dropout rate of 0.2 for the convolutional layers and 0.4 for the dense layers. We clip the ReLU activations at a value of 20. Here, we show the architecture of the character-level model. The word-level model only differs in a larger output dimension (9411 instead of 32). For training we add a CTC loss on top of the dense layer, and for inference a softmax output.

Table 1: Neural network architecture of the character-level model.

Layer	Dimensions [time, freq., channel]	Kernel [time, freq.]	Strides [time, freq.]	Filters
input	$[2500, 40, 1]$			
conv2d	$[1250, 20, 64]$	$[11, 15]$	$[2, 2]$	64
conv2d-1	$[1250, 10, 64]$	$[11, 7]$	$[1, 2]$	64
conv2d-2	$[1250, 5, 192]$	$[11, 7]$	$[1, 2]$	192
reshape-conv2d-to-conv1d	$[1250, 960]$			
conv1d	$[1250, 256]$	7	1	256
conv1d-1	$[1250, 256]$	7	1	256
conv1d-2	$[1250, 256]$	7	1	256
conv1d-3	$[1250, 256]$	7	1	256
conv1d-4	$[1250, 256]$	7	1	256
conv1d-5	$[1250, 256]$	7	1	256
conv1d-6	$[1250, 256]$	7	1	256
conv1d-7	$[1250, 2048]$	32	1	2048
dense	$[1250, 2048]$			
dense-1	$[1250, 32]$			