Rheinisch-Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik 6
Prof. Dr.-Ing. Hermann Ney

Selected Topics in Human Language Technology and Machine Learning in WS 2021-2022

End-to-End ASR LM integration

Jamshaid Sohail

Matrikelnummer 414 361

December 2021

Supervisor: Simon Berger

Contents

1 Introduction **6**
1.1 Automatic Speech Recognition 6
1.2 Language Modelling 6

2 Overview of Automatic Speech Recognition Systems **6**
2.1 Conventional Hidden Markov Based Automatic Speech Recognition 6
2.2 End to End Speech Recognition with Encoder-Decoder systems 7
2.3 Connectionist Temporal Classification 7
2.4 Recurrent Neural Network Transducer (RNN-T) 8
2.5 Attention-based Encoder-Decoder 9
2.6 RNN-AED 9
2.7 Transformer-AED 9

3 What is language model integration and why we need it? **10**

4 External Language Model Integration **10**
4.1 Shallow Fusion 10
4.2 Deep Fusion 10
4.3 Cold Fusion 11
4.4 Component Fusion 12
4.5 Density Ratio Method 12
4.6 Language Model fusion in MWER training 13
4.7 Result And Analysis 14
4.7.1 Cold Fusion 14
4.7.2 Component Fusion 14
4.7.3 Density Ratio Method 16
4.7.4 Language Model Fusion with MWER training 16

5 Internal Language Model Integration **18**
5.1 Hybrid Autoregressive Transducer (HAT) 18
5.2 Internal Language Model Estimation 19
5.3 Internal Language model Training 20
5.4 Result And Analysis 21
5.4.1 Hybrid Autoregressive Transducer 21
5.4.2 Internal Language Model Estimation 21
5.4.3 Internal Language Model training 23

Literaturverzeichnis **25**

List of Tables

1 RNN Language Models with Cold Fusion 14
2 Cold fusion Results 14
3 LM trained on code-switching dataset for Component Fusion 15
4 Component Fusion Results 15
5 Density Ratio Results 16
6 MWER training with SF results 17

7 WER for the baseline and the HAT model 21
8 Out of Domain Results for ILME 22
9 Intra Domain Results for ILME 22
10 RNN-T Results(ILMT) 23
11 AED Results(ILMT) 24

List of Figures

1 RNN-T Model 8

1 Introduction

1.1 Automatic Speech Recognition

ASR involves the process of transcribing an audio into the corresponding text. ASR plays a key role in the field of industrial robotics, medical assistance, defense and aviation and voice assistants. The famous paradigms for establishing the automatic speech recognition systems are the ***Conventional Hidden Markov Model Based systems*** and ***all neural end-to-end speech recognition*** systems. The performance of an ASR system is evaluated using the $word\,error\,rate(WER)$ which is the levenshtein distance between the recognition output and the actual transcription, divided by the number of words in the transcription.

1.2 Language Modelling

A language model estimates the probability $p(w_1^N)$ of a sequence of tokens or words $w_1^N = w_1, ..., w_N$. The language model has its own vocabulary which is a set of token classes on which the model's output distribution is normalized. The probability $p(w_1^N)$ is factorized by the chain rule of probability as below.

$$p(w_1^N) = \prod_{n=1}^{N} p(w_n|w_0^{n-1}) \qquad \text{where } w_0 \text{ is the start symbol of the sequence} \tag{1}$$

Language modelling refers to the problem of predicting the next token or word w_n given the predecessor tokens w_0^{n-1} and it involves finding the conditional probabilities $p(w_n|w_0^{n-1})$. There are two ways in which we try to estimate the probability of the next token given the previous tokens : *n-gram count based language models* and the *neural network based language models*. The evaluation measure for a language model is *perplexity*[10]. It is defined as the inverse geometric average of the conditional probabilities for each word given its predecessor context, according to the language model. The perplexity of the sequence $p(w_1^n)$ for a language model $p(.|.)$ is computed as

$$Perplexity = \frac{1}{\sqrt[N]{(\prod_{n=1}^{N} p(w_n|w_0^{n-1})}} \tag{2}$$

2 Overview of Automatic Speech Recognition Systems

2.1 Conventional Hidden Markov Based Automatic Speech Recognition

We represent the input signal by a sequence of acoustic feature vectors x_1^T and the task of Automatic speech recognition system is to find the most probable word sequence w_1^N corresponding to the input audio speech. The conventional automatic speech recognition system is based on the following Bayes Decision rule.

$$x_1^T \rightarrow \hat{w}_1^N(x_1^T) = \underset{w_1 \ldots w_N}{\operatorname{argmax}} \left\{ p(w_1^N|x_1^T) \right\} \quad \text{with} \quad p(w_1^N|x_1^T) = \frac{p(x_1^T|w_1^N) \cdot p(w_1^N)}{p(x_1^T)} \quad . \tag{3}$$

We can decompose the above equation as below.

$$x_1^T \rightarrow \hat{w}_1^N(x_1^T) = \underset{w_1 \ldots w_N}{\operatorname{argmax}} \left\{ p(w_1^N).p(x_1^T|w_1^N) \right\} \quad . \tag{4}$$

This decomposition has been done using the Bayes Rule Equation in (3). We have omitted the denominator as our acoustic feature vector is fixed. The above decomposition dissociates the language model $p(w_1^N)$, which is trained using the text only data, from acoustic model $p(x_1^T|w_1^N)$ for which a generative model based on Gaussian Mixture Models was available in the statistical modelling. The equation (4) naturally introduces a language model into speech processing system. The decomposition in (4) is also referred to as noisy-channel decomposition.

Features are extracted from the audio signal. *Feature Extraction* involves the process of taking the raw time speech signal and transforming it into *acoustic feature vectors* $p(w_1^N)$ via the signal processing pipeline such as MFCC [6]. The acoustic model, which is $p(x_1^T|w_1^N)$, is based on Hidden Markov Model whose hidden states allows to model the variability in speaking rate of the speaker, and it do so by introducing the concept of alignment between the acoustic feature vector $p(x_1^T)$ and the word sequence w_1^N. This is also referred to as *stochastic modelling* for speech processing.

2.2 End to End Speech Recognition with Encoder-Decoder systems

In the end to end models we use a single neural network to parametrize $p(w_1^N|x_1^T)$ in (3) without decomposition in (4) which involve directly mapping the frame-level input audio features to the output word sequence. These encoder-decoder sequence models jointly learn the acoustic model, pronunciation model and the language model in a single neural network thus giving it the name *end-to-end models*. The emergence of these methods for speech recognition came out from the success of sequence to sequence methods in Machine Translation. The models can be trained to output character-based sub-word units which are called graphemes, byte-pair encodings (BPE) or word pieces without using a hand-crafted pronunciation lexicon.

2.3 Connectionist Temporal Classification

CTC is a training criteria designed for sequence labelling problems like automatic speech recognition where the alignment between the input and the output labels are unknown. It works by adding an extra blank symbol to the target vocabulary and maximize the probabilities of all possible alignments. In case of the cross-entropy loss, the input signal needs to be segmented into words or sub-words, while using the CTC loss it suffices to provide one label sequence for the input sequence and the network learns both the alignment and the labelling. CTC loss is calculated by summing over the probability of possible alignments of input to target. For a given pair of input and output it is calculated as

$$p(w_1^N|x_1^T) = \sum_{y_1^T : a_1^S} \prod_{t=1}^{T} p_t(y_t \mid x_1^T) \tag{5}$$

y_1^T is the alignment sequence with blanks and a_1^S are the true labels without any blanks. CTC assumes the output to be independent of each other and often produce outputs that

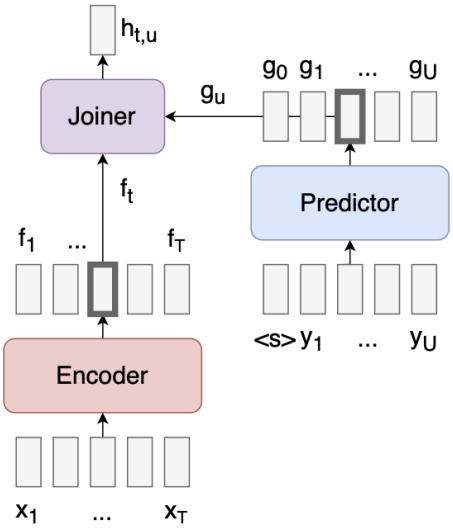

Figure 1: RNN-T Model
[12]

are wrong like "I eight food" instead of "I ate food". Getting good results with CTC requires a secondary language model.

2.4 Recurrent Neural Network Transducer (RNN-T)

Recurrent Neural Network Transducers were designed to overcome the shortcomings of CTC model which required an external language model to perform well. The RNN-T model consists of an encoder, the decoder also called predictor and a joiner or joint network. The encoder converts the acoustic features x_1^T into high level representation $h_{1:T}^{enc}$. The decoder also produces a high level representation h_u^{pre} by consuming previous non-blank target y_1^{u-1} where u is the output label index. The joint network is a feed forward network which combines the encoder network output $h_{1:T}^{enc}$ and the decoder network output and generates a joint matrix $h_{t,u}$ which is used in the softmax output where t denotes the time index.

The encoder network can be uni-directional RNN-LSTM, bi-directional RNN-LSTM, transformer or conformer. If it is uni-directional RNN-LSTM then RNN-T works in the streaming mode and if it is bi-directional RNN-LSTM then RNN-T works in non-streaming mode. When encoder is a uni-directional RNN-LSTM the encoder output is given as

$$h_t^{enc} = LSTM(x_t, h_{t-1}^{enc}) \tag{6}$$

When the encoder is a bi-directional RNN-LSTM then the output of encoder is given as

$$h_t^{enc} = [LSTM(x_t, h_{t-1}^{enc}), LSTM(x_t, h_{t+1}^{enc})] \tag{7}$$

When we implement the prediction network or predictor with a RNN-LSTM the output is

$$h_u^{pre} = LSTM(y_1^{u-1}, h_{u-1}^{pre}) \tag{8}$$

LSTM-RNN in encoder can be replaced with Transformer to construct Transformer Transducer [20] and Conformer Transducer [7].

2.5 Attention-based Encoder-Decoder

Attention based models have been successfully applied in the field of Automatic Speech Recognition. Their success in Neural Machine Translation motivated the researchers to apply them in speech recognition. In speech recognition, attention based models consist of an encoder which encodes the input acoustic speech into a high level representation and an attention based decoder which predicts the next output symbol conditioned on the sequence of previous predictions. The skip-connections are used which allow the information and gradients to flow effectively in the neural network.

2.6 RNN-AED

RNN-AED also consists of an encoder and the decoder network. The encoder of RNN-AED is same as RNN-T model as in equation (6) and (7). Now the decoder is attention enhanced as indicated in the below equation.

$$h_u^{dec} = LSTM(c_u, y_{u-1}, h_{u-1}^{dec}) \tag{9}$$

Where c_u is the context vector obtained by the weighted combination of the encoder output. c_u is expected to have the acoustic information which is used to emit the next token. It is calculated with the help of attention mechanism as in [3, 4].

2.7 Transformer-AED

Transformer based models [11] can handle the long-term dependencies more effectively as the attention mechanism can see all context directly. They are good at capturing content-based global interactions. They do not process the signal sequentially, instead they rely on self-attention to learn the temporal correlations among the sequential signals. One of the big benefits of transformers is that it is simpler to parallelize the computations which can reduce the time to train neural networks on large scale datasets. The encoder is composed of a stack of Transformer blocks. A transformer block consists of multi-head self-attention layer and a feed forward layer. The input of a transformer block is linearly transformed to **Q**, **K** and **V**. The output of Multi-head self-attention layer is

$$Multihead(Q, K, V) = [H_1, ..., H_{d_{head}}]W^{head} \tag{10}$$

$$where\, H_i = softmax(\frac{Q_i K_i^T}{\sqrt{d_k}}) * V_i \tag{11}$$

$$Q_i = QW^{Q_i}, K_i = KW^{K_i} \tag{12}$$

$$V_i = VW^{V_i} \tag{13}$$

Here d_{head} is the number of attention heads and d_k is the dimension for the feature vector for each head. The output is then fed to the feed-forward network. The transformer decoder also has a layer which performs multi-head attention over the output of the encoder.

3 What is language model integration and why we need it?

The role of a language model in speech processing is to estimate the likelihood of a word sequence $w_1, ..., w_N$ to form a valid sentence. Language models can be n-gram/probabilistic language models or neural language models. As we have seen above that encoder-decoder models do not make use of Bayes theorem in (4) which would result in separate language model and acoustic model. Therefore, the encoder-decoder model do not need language model by construction. The decoder part of encoder-decoder model already disposes some *internal language model*. If we want to use a separate language model with a encoder-decoder model then this is referred to as *external language model*. Such external language models can be trained with huge amount of unstructured text data out there without requiring labelled audio signal data with human transcriptions. Integrating an external language model with sequence-to-sequence model helps improve the fluency of generated text. The decoder acts as a internal language model and is trained using transcribed audio data and may not be sufficiently exposed to rare words and phrases thus requiring a need for a external language model.

4 External Language Model Integration

4.1 Shallow Fusion

The idea of shallow fusion was first introduced in context of Machine Translation in [8] and then it was applied for Automatic Speech Recognition systems. In shallow fusion, the external language model is incorporated by the log-linear interpolation of the scores from the sequence to sequence model and the language model at the inference time. Both the sequence to sequence model and the language model are trained independently and then their scores are interpolated at the probability level. The mathematical form of the shallow fusion is given as below.

$$\hat{y} = \underset{w_1^N}{\operatorname{argmax}}(logp_{_{ASR}}(w_1^N|x_1^T) + \lambda logp_{_{LM}}(w_1^N)) \tag{14}$$

where λ is a tunable parameter to define the importance of the external language model. It is usually tuned on the development set. $p_{_{LM}}(w_1^N)$ is the language model probability assigned to the word sequence w_1^N. $logp_{_{ASR}}(w_1^N|x_1^T)$ is calculated from the sequence to sequence model and $p_{_{LM}}(w_1^N)$ is calculated from the language model. x_1^T is an input acoustic sequence and $\hat{y}$ is the predicted label sequence selected among all w_1^N.

4.2 Deep Fusion

The idea of Deep Fusion was also first introduced in context of machine translation in [8]. It showed improvement over the shallow fusion. It fuses the hidden states of sequence to sequence decoder and a neural language model with a gating mechanism, after the two

models are trained independently. In deep fusion, the parameters of the gating part are fine-tuned after parameters of both the sequence to sequence model and language model are frozen. It is also referred to as *fine-tuning fusion* in some literature. The mathematical equations for the deep fusion are as follows.

$$g_t = sigmoid(U_g s_t^{LM} + b), \tag{15}$$

$$\hat{h_t}^{att} = [h_t^{att}; g_t s_t^{LM}], \tag{16}$$

$$y_t = softmax(W_o^{'} \hat{h_t}^{att}) \tag{17}$$

Here g_t is the gate output which is parametrized by U_g which controls the importance of the contribution of the hidden state of the language model LM s_t^{LM}. The concatenated hidden state output $\hat{h_t}^{att}$ is then used to predict the target label using the softmax function. The final step in deep fusion is to fine-tune the combining parameters U_g and $W_o^{'}$ using a small dataset.

4.3 Cold Fusion

Deep Fusion has some limitations which Cold fusion overcomes. The task specific sequence to sequence model in deep fusion is trained independently from the language model. The residual language model which is trained in the decoder part of the sequence to sequence model is biased towards the training corpus. For example, if we train a sequence to sequence model on the legal documents and then later on fuse a language model which is trained on medical documents then the decoder has a tendency to follow the linguistic structure found in legal documents. Cold fusion encourages the decoder part of sequence to sequence model to use the external language model during training [1]. This also helps in faster convergence and better generalization and enables the transfer to a new domain. In cold fusion the sequence to sequence model is trained from scratch together with a pre-trained language model. Now the parameters of the language model are kept frozen and parameters of sequence to sequence models are not. This is in contrast to Deep Fusion where the parameters of both the sequence to sequence and the language models are kept frozen.

The authors in [1] employ in the following architecture. The author uses both the sequence to sequence hidden state s_t and the language model hidden state s_t^{LM} as inputs to the gate computation. The author employs in fine-grained gating mechanism as introduced in [9]. At the last the authors replace the language model hidden state with the language model probability. Then they present the following working of the cold fusion.

$$h_t^{LM} = DNN(l_t^{LM}), \tag{18}$$

$$g_t = sigmoid(W[s_t; h_t^{LM}] + b), \tag{19}$$

$$s_t^{CF} = [s_t; g_t \odot h_t^{LM}] \tag{20}$$

$$r_t^{CF} = DNN(s_t^{CF}) \tag{21}$$

$$\hat{P}(y_t|x, y_{<t}) = softmax(r_t^{CF}) \tag{22}$$

Here l_t^{LM} is the logit output of the language model, s_t is the state of the task specific model and s_t^{CF} is the final fused state which is used to generate the output. DNN can be a deep neural network with any number of layers. The authors found out that single affine layer and ReLU activation prior to softmax activation was helpful.

4.4 Component Fusion

The idea of component fusion was proposed by the authors in [17]. It also incorporates an externally trained language model into an attention based sequence to sequence model. The authors equip the attention-based system with an additional Language model component which can be replaced by an externally trained neural network based language model at inference. The external language model is trained on training transcriptions instead of external text corpus. The whole purpose of doing so is to reduce the mismatch between the external language model and decoder. External language model converges faster and also gives better performance due to this training. The authors let the language model to impact the training of the sequence to sequence model at an early stage by concatenating the gated language model output with the output of the decoder h^{dec} instead of the attentional output.

$$h_t^{dec} = [h_t^{dec}; g_t h_t^{LM}] \tag{23}$$

4.5 Density Ratio Method

Density Fusion Method was proposed by authors in [13]. Density ratio method can be seen as an extension of shallow fusion method. It offers a theoretical grounding in Bayes Rule. This method is an application of Bayes Rule to end-to-end systems and separate language models. The authors denote the source domain by ψ and the target domain by τ. It makes the following assumptions. The source domain ψ has some true joint distribution $P_\psi(w_1^N, x_1^T)$ over text and audio. The target domain τ has some other true joint distribution $P_\tau(w_1^N, x_1^T)$. A source domain end-to-end model for example RNN-T captures $P_\psi(w_1^N|x_1^T)$ reasonably well. The Language models which are separately trained (e.g RNNLM) capture $P_\psi(w_1^N)$ and $P_\tau(w_1^N)$ reasonably well. $P_\psi(x_1^T|w_1^N)$ is roughly equal to $P_\tau(x_1^T|w_1^N)$ i.e the two domains are acoustically consistent. The target domain posterior $P_\tau(w_1^N|x_1^T)$ is unknown. **Density Ratio Method** then expresses a hybrid scaled acoustic likelihood for the source domain.

$$p_\psi(x_1^T|w_1^N) = p_\psi(x_1^T)\frac{P_\psi(w_1^N|x_1^T)}{P_\psi(w_1^N)} \tag{24}$$

Also for the target domain.

$$p_\tau(x_1^T|w_1^N) = p_\tau(x_1^T)\frac{P_\tau(w_1^N|x_1^T)}{P_\tau(w_1^N)} \tag{25}$$

Based on the above stated assumptions the authors calculate the target domain posterior as.

$$\hat{P}_\tau(w_1^N|x_1^T) = k(X)\frac{P_\tau(w_1^N)}{P_\psi(w_1^N)}P_\psi(w_1^N|x_1^T) \tag{26}$$

Here $k(X) = \frac{p_\phi(x_1^T)}{p_\tau(x_1^T)}$ shared by all hypotheses w_1^N, and the ratio $\frac{P_\tau(w_1^N)}{P_\psi(w_1^N)}$ which is the probability mass ratio gives the Density Ratio method it's name. The author shows of

application of Density Ratio Method to RNN-Transducers. (26) can be implemented via an RNN-Transducer "pseudo-posterior" when s_{i+1} is a non-blank symbol.

$$\hat{P}_\tau(s_{i+1}|x_1^T, t_i, s_{1:i}) = \frac{P_\tau(s_{i+1}|s_{i:1})}{P_\psi(s_{i+1}|s_{i:1})} P_\psi(s_{i+1}|x_1^T, t_i, s_{i:1}) \tag{27}$$

The authors used the scaling factors λ_ψ and λ_τ on the language model scores and a non-blank reward β are used in the final decoding score.

$$Score(s_{i+1}|x_1^T, t_i, s_{1:i}) = \log P_\psi(s_{i+1}|x_1^T, t_i, s_{1:i}) + \lambda_\tau \log P_\tau(s_{i+1}|s_{1:i}) \tag{28}$$

$$-\lambda_\tau \log P_\psi(s_{i+1}|s_{1:i}) + \beta \tag{29}$$

4.6 Language Model fusion in MWER training

The authors in [15] proposed a method for language model fusion in MWER training of an E2E model to discard the need for LM weights tuning during inference. MWER stands for minimum word error rate and MWER training aims to mitigate the mismatch between the training criteria and the evaluation metric of a speech recognition model. MWER minimizes the expected word error. With the help of MWER training, the E2E model is further fine-tuned to directly minimize the expected number of word errors on the training corpus. The authors proposed MWER with shallow fusion(MWER-SF) and MWER with ILME (MWER-ILME). MWER-ILME involves the internal language model estimation with minimum word error rate training. We will be looking into ILME (Internal Language Model Estimation) in more detail in the internal language model integration methods.

In [16] the authors perform shallow fusion during MWER training to generate N-best hypotheses $Y^1, ..., Y^N$ of x_1^T. Linear combination of E2E model and the LM probabilities helps in calculating the posterior of each hypothesis. In contrast to this calculation, the authors in the current paper obtained the hypotheses posterior by the interpolation of log-probabilities of E2E model and the external LM. The MWER-SF loss is given by the authors as below.

$$L_{MWER}^{SF} = \Sigma_{n=1}^N \bar{P}(w_1^N|x_1^T; \theta_{E2E}^S, \theta_{LM}^T) R(w_1^N, Y^*) \tag{30}$$

Here $\bar{P}(w_1^N|x_1^T; \theta_{E2E}^S, \theta_{LM}^T)$ is the re-normalized shallow fusion probability over N-best hypotheses. $R(w_1^N, Y^*)$ is the number of word errors compared to the reference Y^*.

In the same way the authors perform ILME based fusion in MWER training. It differs from the simple MWER training in the way that authors apply ILME-based fusion to generate N-best hypotheses of training utterances and the authors compute hypotheses posteriors using probabilities of E2E model, internal language model and the external language model. The MWER-ILME loss function is given as

$$L_{MWER}^{ILME} = \Sigma_{n=1}^N \bar{P}(w_1^N|x_1^T; \theta_{E2E}^T) R(w_1^N, Y^*) \tag{31}$$

MWER-ILME training is minimizing the expected number of word errors over the N-best hypotheses generated by the target domain E2E model θ_{E2E}^T. MWER-ILME helps adapt the E2E model towards a fixed external LM along with internal LM weight λ_s and external language model weight λ_T.

4.7 Result And Analysis

4.7.1 Cold Fusion

The author used two datasets as the *source* domain and the *target* domain datasets. The source domain dataset was search queries (411,000 utterances/650 hours of audio) and the target domain dataset was movie scripts (345,000 utterances /676 hours of audio). The author then trained different language models out of these datasets, whose results came out to be as below. Here Full refer to the mixture of the source and target dataset.

Table 1: RNN Language Models with Cold Fusion

Model	Domain	Word Count	Perplexity Source	Perplexity Target
GRU (3x512)	Source	5.73M	2.6790	4.463
GRU (3x512)	Target	5.46M	3.717	2.794
GRU (3x1024)	Full	25.16M	**2.491**	**2.325**

The neural network architecture consisted of an encoder and a decoder. The soft-attention mechanism was applied as in [2]. The encoder consisted of 6 BLSTM (Bi-directional LSTM) layers each having a dimensions of 480. The authors also added max-pooling layers with a stride of 2 along the time dimension after the first 2 BLSTM layers and added residual connections for each of the BLSTM layers to speed up the training process. The decoder employed in hybrid attention as in [5] and consisted of a single layer of a 960 dimensional gated recurrent unit (GRU). The author trained the entire system with Adam optimizer with a batch size of 64. The learning rates were tuned separately for each model using random search. The authors used a fixed beam size of 128 for each of the experiments. The authors used schedule sampling with a sampling rate of 0.2 which was fixed during the whole training. Schedule sampling helped reduce the effect of exposure bias due to the differences in training and inference mechanisms. WER stands for word error rate in the below table and it is a metric to measure the performance of the automatic speech recognition. It is calculated by dividing the number of errors(substitutions + insertions + deletions) by the total number of words spoken.

Table 2: Cold fusion Results

Model	Train Domain	Test on Source WER(%)	Test on target WER(%)
Baseline Attention Model	Source	14.68	43.52
Baseline Attention Model	Target		17.61
Baseline + Deep fusion	Source	13.92	37.45
Baseline + Cold fusion	Source	13.88	30.71

4.7.2 Component Fusion

The authors evaluated the component fusion for two scenarios, out-of-domain and in-domain. For the out-of-domain the authors collected two datasets the monolingual man-

darin and the english dataset which serve as the source dataset and the Mandarin-English code-switching dataset which serve as the target dataset. RNNLM was trained on a considerable larger amount of external in-domain and was incorporated into the attention based model. The authors used the public datasets AISHELL-1 and AISHELL-2 for the purpose. The authors used the encoder with 6BLSTM layers each having 1024 LSTM units. The decoder consisted of 2 LSTM layers each having 1024 LSTM units. The authors used the ADAM optimizer with default parameters and default learning rate was set to 0.001. The authors also employed in dropout with probability of 0.2. The external language model consisted of a character-based 3 layer GRU model each having 1024 GRU units and the training procedure was same as above attention model training procedure. The authors got the below results for language modelling using the code-switching test set.

Table 3: LM trained on code-switching dataset for Component Fusion

Model	**Perplexity**
Source	63.02
Target	13.12
source + target	12.46

These neural network language models were built on character level. The authors performed the domain adaptation using the proposed component fusion method. The source Neural Network LM is trained on speech transcriptions and is frozen when attention model is being trained. The author replaces the source NN LM with target NN LM during decoding. The results of using the component fusion for out-of-domain datasets is given below and it can be seen that it presented a better performance. The authors also explored the performance of concatenating the gated LM with the hidden state of the encoder h^{dec}. This allowed the Language model to impact the training of the attention model in an earlier stage. It can be seen in the table below that both component and cold fusion with h^{dec} achieves a better performance. In the below table CER stands for character error rate and now it is calculated at the character level instead of word level like WER.

Table 4: Component Fusion Results

Model	**CER(%)**
baseline + Shallow Fusion	25.01
Cold Fusion (h^{att})	25.37
Cold Fusion (h^{dec}) + Shallow fusion	20.40
Component Fusion (h^{dec}) + Shallow Fusion	17.53

4.7.3 Density Ratio Method

The author trains a end-to-end model such as RNN-T on given source domain training set ψ (paired audio/transcript data). A language model is trained on text transcripts from the same training set ψ. The author then trains a second language model on the target domain τ. When decoding on the target domain, modify the RNN-T output by the ratio of the target/training RNN-LMs as defined in (28).

120M segemented utterances from youtube videos with with associated transcripts were used for source domain baseline RNN-T. Transcripts from the same 120M utterance youtube training set were used for source-domain normalizing RNN-LM. For the target domain RNNLM, 21M text-only utterance-level transcripts from anonymized, manually transcribed audio data, representative of data from a voice search service was used. For the target-domain RNN-T fine-tuning, 10K, 100K, 1M and 21M utterance level pairs taken from anonymized, transcribed voice search data was used. The datasets to choose the optimal scaling factors and evaluate the final model consisted of source domain evaluation set (youtube videos) and target domain evaluation and development sets (Voice search).

The RNN-T consisted 6 LSTM layers encoder which is bidirectional. The decoder consisted of 1 LSTM layer. Acoustic features consisted of 768 dimensional feature vectors obtained from 3 stacked 256-dimensional logmel feature vectors. Output classes consisted of 10000 sub-word morph units and total number of parameters for the network were 340M. RNNLMs for both the source and target domains were set to match the RNN-T decoder size and structure. RNN-T and RNNLMs were independently trained on 128-core tensor processing unit and an effective batch size of 4096. All models were trained using the ADAM optimization. The models were trained and different combination of scaling factors λ_ψ and λ_τ were tried out. The results of the experiments are summarized in the table and it can be seen that density ratio method is beating the shallow fusion.

Table 5: Density Ratio Results

Model	**WER(%)**	λ	β
Baseline (no-fine tuning)	17.5	-	-0.3
21000h fine-tuning	7.8	-	0.1
Shallow fusion	7.7	0.1	0.3
Density Ratio ($\lambda_S = \lambda_T$)	7.4	0.1	0.0

4.7.4 Language Model Fusion with MWER training

The authors trained a streaming transformer-transducer (T-T) models to minimize the transducer, MWER, MWER-SF and MWER-ILME losses. The evaluation was done on multi-domain test data. The generalizability of MWER-SF and MWER-ILME training is checked by evaluating with an out-of-domain LM on an out-of-domain test set. The authors considered two language models multi-domain language models and the librispeech language models. The authors first train a uni-directional long-short term memory LM on 2 billion words of anonymized text data. The authors train another language model with 9.4M word transcript of the 960h training speech and the 813M-word text in Libri-speech corpus.

The authors first train an transformer-transducer with 30K hour data. The baseline T-T is then fine-tuned with the same data to minimize MWER, MWER-SF and MWER-ILME losses using ADAM optimizer with a constant learning rate of 10^{-5}. The authors did multi-domain and the out-of-domain evaluations. The multi-domain evaluations was done using call, meeting, search, keyboard, email and common voice datasets. The performance of E2E models was evaluated by fusing the language models trained on Libri-speech dataset.

Table 6: MWER training with SF results

Test subset	**T-T WER(%)**	**MWER WER(%)**	**MWER-SF WER(%)**
Call	8.70	8.37	7.70
Meeting	16.34	16.07	17.54
Search	12.56	12.35	12.13
Keyboard	7.95	7.56	7.60
Email	19.16	17.67	16.94
Common	12.43	11.75	11.34

5 Internal Language Model Integration

As discussed above, the decoder part of the encoder-decoder model already disposes off some language model which is referred to as *Internal Language Model*. In this section we will be looking into some internal language model integration methods and how they have helped in improving the performance of the end-to-end Automatic Speech Recognition systems.

5.1 Hybrid Autoregressive Transducer (HAT)

The idea of HAT was proposed by authors in [18]. The HAT model provides a way to measure the quality of internal language model that can be used to decide whether inference with an external language model is beneficial or not. HAT is a time-synchronous encoder-decoder model which couples the powerful probabilistic capability of sequence to sequence models with an inference algorithm which preserves the modularity and external lexicon and LM integration. The author also presents a finite history version of HAT.

It is a type of time-synchronous encoder-decoder model which distinguishes itself from the other time-synchronous models by formulating the local posterior probability differently and by providing a measure of its internal language model quality and offering a mathematically justified inference algorithm for an external language model.

The RNN-T lattice definition is used for HAT model. HAT model differentiates between the horizontal and vertical edges in the lattice to calculate the local conditional posterior. For edge k corresponding to position (t, u), the posterior probability to take a horizontal edge and emitting $\langle b \rangle$ is modelled by a Bernoulli Distribution $b_{t,u}$, which is the function of the entire past history of labels and features. The posterior distribution $P_{t,u}(y_{u+1}|x_1^T, y_{0:u})$, defined over labels in V, helps modelling the vertical move. The alignment posterior is formulated as

$$\begin{cases} b_{t,u}, & \tilde{y_k} = \langle b \rangle \\ (1 - b_{t,u}) P_{t,u}(y_{u+1}|x_1^T, y_{0:u}), & \tilde{y_k} = y_{u+1} \end{cases} \tag{32}$$

The input feature sequence x_1^T is fed to a stack of RNN layers to output T encoded vectors $f^1(X) = f_{1:T}^1$. The label sequence is also fed to a stack of RNN layers to output $g^1(Y) = g_{1:U}^1$. The formula of conditional bernoulli distribution $b_{t,u}$ is calculated as below.

$$b_{t,u} = sigmoid(w \cdot (f_t^1 + g_u^1) + b) \tag{33}$$

There are two functions. $f^2(X) = f_{1:T}^2$ encodes input features x_1^T and the function $g^2(Y) = g_{1:U}^2$ encodes label embeddings. At each time position (t,u), the joint score is calculated over all $y \epsilon V$

$$S_{t,u}(y|x_1^T, y_{0:u}) = J(f_t^2 + g_u^2) \tag{34}$$

Here $J(\odot)$ is any function that maps $f_t^2 + g_u^2$ to $|V|$ dimensional score vector. The label posterior distribution is derived as below.

$$P_{t,u}(y_{u+1}|X, y_{0:u}) = \frac{exp(S_{t,u}(y_{u+1}|x_1^T, y_{0:u}))}{\sum_{y \epsilon V} exp(S_{t,u}(y|x_1^T, y_{0:u}))} \tag{35}$$

The alignment path posterior of HAT model is calculated as below.

$$P(\tilde{Y}|x_1^T) = \prod_{k=1}^{T+U-1} P(\tilde{Y}_k|x_1^T, \tilde{Y}_{1:k-1}) \tag{36}$$

The HAT model produces a local and sequence level internal LM scores. The sequence level internal language model score is given as below.

$$logP_{ILM}(Y) = \Sigma_{u=0}^{U-1} logP(y_{u+1}|y_{0:u}) \tag{37}$$

The HAT model inference search for $\tilde{y}^*$ that maximizes.

$$\lambda_1 logP(w_1^N|x_1^T) - \lambda_2 logP_{ILM}(B(w_1^N)) + logP_{LM}(w_1^N) \tag{38}$$

Here B is a function that maps permitted alignment paths to the corresponding label sequences.

5.2 Internal Language Model Estimation

The authors in [19] proposed the idea of Internal Language model estimation ILME which facilitates the more effective integration of an external language with all existing end-to-end models with no additional model training including the RNN-T and the attention based models. With the help of ILME, the internal language model scores of E2E model are estimated and subtracted from the log-linear interpolation between the scores of the E2E model and the external LM. The internal LM scores are estimated as the output of an E2E model when eliminating its acoustic component. ILME has shown the improvements over the shallow fusion method. It has also shown improvements over the density ratio method, despite having less parameters.

Density ratio method assumes that the posterior of a source domain end-to-end model is decomposable into individual acoustic model and language model with separate parameters like a hybrid system. The source domain End-to-end model is factorized as follows using the Bayes theorem.

$$P(Y|X;\theta_{E2E}^S) = \frac{P(X|Y;\theta_{E2E}^S)P(Y;\theta_{E2E}^S)}{P(X;\theta_{E2E}^S)} \tag{39}$$

Here all factors are conditioned on the same set of E2E parameters θ_{E2E}^S and $P(Y;\theta_{E2E}^S)$ is the internal LM of the E2E model. Given constant acoustic conditions i.e $P(X|Y;\theta_{E2E}^S)$ $= P(X|Y;\theta_{AM}^T)$, the target domain end-to-end posterior is calculated as follows.

$$P(Y|X;\theta_{E2E}^T) = P(Y|X;\theta_{E2E}^S)\frac{P(Y;\theta_{LM}^T)}{P(Y;\theta_{E2E}^S)}\frac{P(X;\theta_{E2E}^S)}{P(X;\theta_{E2E}^T)} \tag{40}$$

During inference, the log probability of the internal language model is subtracted from log-linear combination between the scores of the E2E and external language models as follows.

$$\hat{Y} = \underset{Y}{\text{argmax}}[logP(Y|X;\theta^S_{E2E}) + \lambda_T logP(Y;\theta^T_{LM}) - \lambda_I logP(Y;\theta^S_{E2E})] \tag{41}$$

Here λ_I is the internal language model weight. This LM integration method is referred to as Internal Language model estimation (ILME). This differs from the density ratio method in the way that it subtracts the log probability of an E2E internal LM parametrized by θ^S_{E2E} rather than that of a source domain LM separately-trained with the training transcript of the E2E model. The key step of the ILME is to estimate the internal language model below defined as the token sequence probability distribution an E2E model implicitly learns from the audio-transcript training pairs.

$$P(Y;\theta^S_{E2E}) = \prod_{u=1}^{U} P(y_u|Y_{0:u-1};\theta^S_{E2E}) \tag{42}$$

$$= \prod_{u=1}^{U} \prod_{X} P(y_u|X, Y_{0:u-1};\theta^S_{E2E}) P(X|Y_{0:u-1};\theta^S_{E2E}) \tag{43}$$

The summation over the entire acoustical space in above equation is intractable in practice. To address this, Proposition 1 in appendix A of [18] was proposed and proved to approximate the internal LM of a HAT model by eliminating the effect of encoder activations. The authors named this proposition as Joint Softmax Approximation (JSA). The authors apply JSA to estimate the internal language models of the pre-existing most popular end-to-end models.

5.3 Internal Language model Training

The authors in [14] present the above method. This method is meant to improve the Internal Language model estimation. ILMT minimizes an additional LM loss by updating only the E2E model components that affect internal LM estimation. This method encourages the E2E model to form a standalone LM inside its existing components, without sacrificing the ASR accuracy. ILMT with ILME-based inference helps in improving the results. The accuracy of internal LM estimation is not guaranteed if the E2E model is not structured in a way that strictly satisfies the conditions of Proposition 1 in [18] [Appendix A].

The authors jointly minimize an internal language model loss together with a standard E2E loss during ILMT. The internal LM loss of an RNN-T model is obtained by summing up the negative log probabilities of the internal language model over the training corpus as follows.

$$L(\theta_{pred}, \theta_{joint}) = -\Sigma_{Y \epsilon D} \Sigma_{u=1}^{U} logP(y_u|Y_{0:u-1};\theta_{pred}, \theta_{joint}) \tag{44}$$

For RNN-T, the ILMT loss is constructed as a weighted sum of the RNN-T loss and the ILM loss below.

$$L_{ILMT}(\theta_{RNN-T}) = L_{RNN-T}(\theta_{RNN-T}) + \alpha L_{ILM}(\theta_{pred}, \theta_{joint}) \tag{45}$$

Here α is the weight of the internal language model loss. Minimizing the RNN-T ILMT loss, we maximize the internal LM probability of the E2E training transcripts by updating only the prediction and the joint networks. The internal LM loss of AED is formulated as a summation of negative log probabilities of the internal LM over training corpus.

$$L_{ILM}(\theta_{dec}) = -\Sigma_{Y \epsilon D} \Sigma_{u=1}^{U+1} logP(y_u|Y_{0:u-1}; \theta_{dec}) \tag{46}$$

For AED, the ILMT loss is computed as a weighted sum of the AED loss and the ILM loss below.

$$L_{ILMT}(\theta_{AED}) = L_{AED}(\theta_{AED}) + \alpha L_{ILM}(\theta_{dec}) \tag{47}$$

5.4 Result And Analysis

5.4.1 Hybrid Autoregressive Transducer

The dataset consisted of training set, development set and the test set. The training consisted of 40M utterances, development set consisted of 8K utterances and test set consisted of 25 hours. All the datasets were anonymized and representative of Google Traffic Queries. All models (baselines and HAT) were trained to predict 42 phonemes and are decoded with a lexicon and a n-gram language model that cover a 4M words vocabulary.

The author presented three time synchronous baselines. The encoder consisted of 5 layers of LSTM with 2048 cells per layer. For the models which consisted of a decoder network has 2 layers of LSTM with 256 cells per layer. The authors observe that for the inference algorithm in (38) the values $\lambda_1 \in (2.0, 3.0)$ and $\lambda_2 = 0.1$ lead to the best performance. The HAT model outperforms the baseline RNNT model by 1.4% absolute WER.

Table 7: WER for the baseline and the HAT model

Model	WER(del/ins/sub)
CTC	8.9(1.8/1.6/5.5)
RNN-T	8.0(1.1/1.5/5.4)
HAT	6.6(0.8/1.5/4.3)

5.4.2 Internal Language Model Estimation

The authors evaluated the models for both cross-domain and intra-domain scenario. The testing was done for both RNN-T and the AED model. The authors trained a RNN-T and an AED model with 30k hours of anonymized and transcribed data from microsoft services. Both RNN-T and AED models employed in LSTM model in their different components. The results of the out-of-domain and the in-domain experiments for both the RNN-T and AED models came out to be as below.

Table 8: Out of Domain Results for ILME

E2E Model	**Methods**	**Params**	λ_T	μ	**WER Dev**	**WER Test**
RNN-T	BS	76M	-	-	20.03	20.23
RNN-T	SF	134M	0.07	0.00	18.37	18.88
RNN-T	DR	191M	0.20	0.12	16.14	18.07
RNN-T	ILME	134M	0.24	0.12	15.48	17.01
AED	BS	97M	-	-	18.10	22.04
AED	SF	155M	0.10	0.00	12.84	13.39
AED	DR	212M	0.12	0.02	12.22	12.86
AED	ILME	155M	0.13	0.10	11.72	12.31

Table 9: Intra Domain Results for ILME

E2E Model	**Methods**	**Params**	λ_T	μ	**WER**
RNN-T	BS	76M	-	-	16.16
RNN-T	SF	134M	0.03	0.00	15.77
RNN-T	DR	191M	0.21	0.19	15.64
RNN-T	ILME	134M	0.26	0.20	14.70
AED	BS	97M	-	-	14.08
AED	SF	155M	0.09	0.00	12.96
AED	DR	212M	0.09	0.05	12.89
AED	ILME	155M	0.13	0.08	12.36

5.4.3 Internal Language Model training

The procedure of ILMT with ILME-based inference for the LM integration with an E2E model is the following. Train an E2E model with the source domain audio transcript pairs to minimize the ILMT loss for AED or RNN-T. Train an external LM with with the target domain-only text data. Integrate the ILMT E2E model in first step with the LM trained in second step using the ILME based inference.

The dataset used for ILMT E2E training is the 30k hour of anonymized and transcribed data collected from Microsoft services. The encoder and prediction networks are both uni-directional LSTMs with 6 and 2 hidden layers respectively and 1024 hidden units in each layer. The joint network has 4000 dimensional output units. The RNN-T has 76M parameters. For AED, the encoder is a bi-directional LSTM with 6 hidden layers and 780 hidden units in each layer. The decoder is a uni-directional LSTM with 2 hidden layers each with 1280 hidden units. The decoder has 4000-dimensional output units. The authors perform both the cross-domain and the intra-domain(Dictation test set and conversation test set) evaluations. For the cross-domain, authors used Librispeech clean test set by integrating an LSTM-LM trained with librispeech text. The overall results with RNN-T and AED models are listed in the below tables.

Table 10: RNN-T Results(ILMT)

Train loss	**Evaluation Method**	**Model Params**	**Librispeech Dev WER**	**Librispeech Test WER**
L_{RNN-T}	No LM	76M	9.27	8.97
L_{RNN-T}	Shallow Fusion	134M	7.44	7.53
L_{RNN-T}	Density Ratio	191M	6.80	6.74
L_{RNN-T}	ILME	134M	6.41	6.36
L_{ILMT}	No LM	76M	8.58	8.37
L_{ILMT}	Shallow Fusion	134M	6.60	6.47
L_{ILMT}	Density Ratio	191M	5.86	5.61
L_{ILMT}	ILME	134M	5.57	5.30

Table 11: AED Results(ILMT)

Train loss	**Evaluation Method**	**Model Params**	**Librispeech Dev WER**	**Librispeech Test WER**
L_{AED}	No LM	97M	8.56	8.61
L_{AED}	Shallow Fusion	155M	5.00	5.33
L_{AED}	Density Ratio	212M	4.74	5.09
L_{AED}	ILME	155M	4.42	4.87
L_{ILMT}	No LM	97M	7.31	7.47
L_{ILMT}	Shallow Fusion	155M	6.54	6.61
L_{ILMT}	Density Ratio	212M	4.28	4.85
L_{ILMT}	ILME	155M	3.30	3.65

It can seen from the above tables that all LM integration methods consistently achieve lower WERs with ILMT than with standard E2E training. Among all methods ILMT with ILME-based inference gives the best results.

References

[1] S. Satheesh A. Sriram, H. Jun and A. Coates. Cold fusion: Training seq2seq models together with language models. In *Interspeech2018*.

[2] Chorowski Jan Serdyuk Dmitriy Brakel Philemon Bahdanau, Dzmitry and Yoshua Bengio. End-to-end attention-based large vocabulary speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4945–4949, IEEE, 2016.

[3] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *arXiv preprint arXiv:1409.0473*, 2014.

[4] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio. End-to-end attention-based large vocabulary speech recognition. In *Proc. ICASSP*, pages 4945–4949, IEEE, 2016.

[5] Chorowski, Jan K, Bahdanau, Dzmitry, Serdyuk, Dmitriy, Cho, Kyunghyun, and Yoshua Bengio. Attention-based models for speech recognition. In *Advances in Neural Information Processing Systems*, pages 577–585, 2015b.

[6] S. David and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. In *IEEE Transactions on Acoustics, Speech, and Signal Processing*, volume 87, pages 1738–1752, 1980.

[7] A. Gulati, J. Qin, and C.-C. Chiu. Conformer: Convolutionaugmented transformer for speech recognition. In *arXiv preprint arXiv:2005.08100*, 2020.

[8] Firat Orhan Xu Kelvin Cho Kyunghyun Barrault Loic Lin Huei-Chi Bougares Fethi Schwenk Holger Gulcehre, Caglar and Yoshua Bengio. On using monolingual corpora in neural machine translation. In *arXiv preprint arXiv:1503.03535*, 2015.

[9] Zhang Xiangyu Ren Shaoqing He, Kaiming and Jian Sun. Deep residual learning for image recognition. In *CoRR*, abs/1512.03385, 2015.

[10] F. Jelinek, R.L. Mercer, L.R. Bahl, and J.K. Baker. Perplexity measure of the difficulty of speech recognition tasks. In *The Journal of the Acoustical Society of America,*, volume 62, 1977.

[11] Jinyu Li, Yashesh Gaur Yu Wu, Chengyi Wang, Rui Zhao, and Shujie Liu. On the Comparison of Popular End-to-End Models for Large Scale Speech Recognition. In , 2020.

[12] Loren Lugosch. Sequence-to-sequence learning with Transducers. , November 2020.

[13] Erik McDermott, Hasim Sak, and Ehsan Variani. A Density Ratio approach to Language Model Fusion in end-to-end Automatic Speech Recognition. In *Proc. ASRU*, page 434441, February IEEE, 2019.

[14] Zhong Meng, Naoyuki Kanda, Yashesh Gaur, Sarangarajan Parthasarathy, Eric Sun, Liang Lu, Xie Chen, Jinyu Li, and Yifan Gong. Internal language model training for domain-adaptive end-to-end speech recognition. In *Proc. ICASSP*, IEEE, 2021.

[15] Zhong Meng, Yu Wu, Naoyuki Kanda, Liang Lu, Xie Chen, Guoli Ye, Eric Sun, Jinyu Li, and Yifan Gong. Minimum Word Error Rate Training with Language Model Fusion for End-to-End Speech Recognition. In *INTERSPEECH*, 2021.

[16] C. Peyser, S. Mavandadi, and T. Sainath et al. Improving tail performance of a deliberation e2e asr model using a large text corpus. In *INTERSPEECH*, 2020.

[17] Changhao Shan, Chao Weng, Guangsen Wang, Dan Su3, Min Luo2, Dong Yu4, and Lei Xie1y. Component Fusion: Learning Replaceable Language Model Component for End-to-end Speech Recognition System. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5361–5635, 2019.

[18] Ehsan Variani, David Rybach, and Michael Riley Cyril Allauzen. Hybrid Autoregressive Transducer (HAT). In *Proc. ICASSP*, page 61396143, IEEE, 2020.

[19] E. Sun et al. Z. Meng, S. Parthasarathy. Internal language model estimation for domain-adaptive end-to-end speech recognition. In *Proc. SLT*, IEEE, 2021.

[20] Q. Zhang, H. Lu, and H. Sak. Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss. In *Proc. ICASSP*, 2020.