UNSUPERVISED CONTEXT LEARNING FOR SPEECH RECOGNITION

Assaf Hurwitz Michaely, Mohammadreza Ghodsi, Zelin Wu, Justin Scheiner, Petar Aleksic

Google Inc.
{amichaely,ghodsi,zelinwu,jmscheiner,apetar}@google.com

ABSTRACT

It has been shown in the literature that automatic speech recognition systems can greatly benefit from contextual information [1, 2, 3, 4, 5]. Contextual information can be used to simplify the beam search and improve recognition accuracy. Types of useful contextual information can include the name of the application the user is in, the contents of the user's phone screen, the user's location, a certain dialog state, etc. Building a separate language model for each of these types of context is not feasible due to limited resources or limited amounts of training data.

In this paper we describe an approach for unsupervised learning of contextual information and automatic building of contextual biasing models. Our approach can be used to build a large number of small contextual models from a limited amount of available unsupervised training data. We describe how n-grams relevant for a particular context are automatically selected as well as how an optimal size of a final contextual model is chosen. Our experimental results show great accuracy improvements for several types of context.

Index Terms: speech recognition, language modeling, contextual information

1. INTRODUCTION

Using contextual information to adapt the language model (LM) can significantly improve automatic speech recognition accuracy [1, 2, 3, 4, 5]. On-the-fly rescoring [4] has been used to dynamically adjust LM weights for certain n-grams, based on contextual models available at request time.

Previous work using on-the-fly rescoring [4, 3] assumes the existence of a set of contextually relevant phrases to be biased, and focuses on processing these phrases into n-grams and weights. Some examples of such contextual phrase sets: commands for a certain dialog state of the user's phone, entities based on the user's location, and trending search queries. In all these cases, the entire set of phrases is broken up into n-grams which make up the biasing model.

In this paper we present a method for automatically selecting the n-grams and weights. To select these n-grams, we use a general baseline LM and a corpus of contextually relevant queries. The selected n-grams and weights can then be used to adjust the LM to the desired context using on-the-fly rescoring.

One can view our approach as an alternative to keeping a separate full-sized LM for each context. Keeping multiple LMs is not easily scalable, as they require significant compute resources and maintenance. One alternate approach is Bayesian interpolation [6], which focuses on statically interpolating several LMs while optimizing the interpolation weights per n-gram. However, because the interpolation is static, the LM does not make use of contextual signals available at recognition time. In on-demand-interpolation [5], the LM interpolation weights are set in real time, so the LM truly adapts to context. However, there is a significant cost in latency. Furthermore, this LM is not modular—it is assembled offline, and must be reassembled whenever any of the underlying LMs is changed.

Our method uses only high-impact n-grams, keeping the biasing model small, so the system can easily hold thousands of these models to be used on demand, with only minimal latency added, $\sim$100ms per query. Each model can be changed, removed, or added without having to refresh the entire system.

Furthermore, by focusing only on high-impact n-grams, our method avoids having to deal with low probability n-grams. If an n-gram doesn't appear very often in the training data, it will not be selected, and the general LM score will be used instead. This simplifies the training method, and allows for training effective biasing models using far less data than the amount needed to train full LMs for each context.

Finally, the biasing models described in this paper were all built using unsupervised data. The queries used to train the models were transcribed using an automatic speech recognition system.

We organize the paper as follows. In section 2, we describe the LM biasing framework, and present a method for building the biasing models. In section 3, we describe how the model size is optimized. Finally, in section 4, we describe the experiments conducted and present results.

2. BIASING N-GRAM SELECTION PROCESS

In this section we first give a short description of the LM biasing framework we use: on-the-fly rescoring. We then present the criteria we use for n-gram selection, and our algorithm for finding a near-optimal set of n-grams.

2.1. Language model rescoring framework

We use on-the-fly rescoring, a framework for biasing LMs using n-grams, introduced in [4]. The biasing model consists of the set of n-grams to be adjusted and their weights. Biasing is applied during decoding, and may be triggered based on particular conditions for some queries. Biasing affects the weight of some n-grams, while leaving all other weights unchanged. The biasing n-grams can include any symbol, whether in the main LM's vocabulary or not.

Class based symbols [7, 8] are also supported, both for static classes (e.g. "$TIME", "$DATE", "$FULLPHONENUM"), populated at LM training time, and dynamic classes (e.g. "$CONTACTS", populated at recognition time using the user's contact list [9]).

Typically the biasing model is much smaller than the general LM: While state-of-the-art main LMs may contain over 10 million n-grams [6], the biasing models evaluated in this paper are much smaller, ranging from 60 to 10,000 n-grams.

In the context of probabilistic LMs, we assign the cost (a.k.a. score, weight) of a word w, given its history H (the first $n-1$ words in an n-gram), with the negative log probability: $-\log(P(w|H))$.

To find the cost of an n-gram $s(w|H)$ in the combined model, we first try to find the longest non-empty suffix of Hw in the biasing model B, denoted $H'w$. If such a suffix exists, we combine the cost of the general LM, s_{LM} and the cost in the biasing model s_B as:

$$s(w|H) = C(s_{\text{LM}}(w|H), s_B(w|H'))$$

where C is the score combination function. For example, in most of our experiments we use $C = \min$, which provides *positive biasing*[4] toward a context.

If there is no suffix $H'w$ of Hw in B, we simply use the general LM score $s(w|H) = s_{\text{LM}}(w|H)$.

2.2. The criteria for selecting biasing n-grams

Our overall goal is to adapt the LM (using biasing) toward a specific context (represented by a sample set from observed queries), while selecting as few biasing n-grams as possible. We can then override the LM costs of these n-grams to more closely match their observed probabilities in the sample.

One can use perplexity to measure how well an LM matches a given sample data set S drawn from a distribution P_S. For simplicity in the following definitions, we assume S is a sequence of words: $w_1, \dots w_L$. The perplexity of the LM on S is defined as $2^{\text{CE}(P_S, P_{\text{LM}})}$, where $\text{CE}(P_S, P_{\text{LM}})$ refers to the cross-entropy of the sample with respect to the LM (A language model represents a probability distribution over n-grams).

Cross-entropy is defined as

$$\begin{aligned}\text{CE}(P_S, P_{\text{LM}}) &= E_{P_S}\left[-\log P_{\text{LM}}\right] \\ &= -\sum_{Hw} P_{\text{S}}(Hw) \log P_{\text{LM}}(Hw)\end{aligned}$$

Among all possible models (probability distributions), the model that minimizes the above cross-entropy is the sample distribution itself. The difference in cross entropy between any model (e.g. the general LM) and the optimal model is the KL-divergence [10] (D_{KL}) between the model and sample:

$$\begin{aligned}D_{KL}(P_{\text{S}}, P_{\text{LM}}) &= \text{CE}(P_{\text{S}}, P_{\text{LM}}) - \text{CE}(P_{\text{S}}, P_{\text{S}}) \\ &= \text{CE}(P_{\text{S}}, P_{\text{LM}}) - \text{Entropy}(P_{\text{S}}) \\ &\approx \sum_{Hw \in \text{S}} P_{\text{S}}(Hw) \log \frac{P_{\text{S}}(w|H)}{P_{\text{LM}}(w|H)}\end{aligned}$$

We want to replace the probability of a subset of n-grams in the LM with the observed probability in the sample, to minimize the perplexity$(\text{S}, \text{LM+B})$, where LM+B represents the biased LM. Therefore we should pick the n-grams that most decrease D_{KL}.

If the effect of including each n-gram in the biasing model were independent from the other selected n-grams, we could simply sort all n-grams by their contribution to D_{KL}, namely

$$P_{\text{S}}(Hw)\left(\log P_{\text{S}}(w|H) - \log P_{\text{LM}}(w|H)\right)$$

and pick the top k.

Due to the way our biasing framework uses n-grams, their effects are not independent. Specifically, when scoring an n-gram Hw, as long as the biasing model contains some suffix $H'w$ of Hw, the longest such n-gram will be used to override the main LM's cost, even if the main LM has an n-gram of higher order (length of history). For example, if the biasing model consists of a single unigram w, it will override the LM cost for all n-grams Hw regardless of their order.

Because of this dependence, we consider n-grams in order of length. We first decide which unigrams to include in the biasing model. Then, given our previous decisions, decide which bigrams to include, and so on up to the maximum length considered. (n-grams of the same length are independent).

For each n-gram Hw under consideration, we first compare its sample probability to the sample probability of the longest selected n-gram $H'w$ that is its suffix. If no such n-gram $H'w$ has been selected, then instead we compare Hw's sample probability to its probability in the main LM. We denote this value with $\Delta_{\text{adapt}}(Hw, LM, B)$:

$$\begin{aligned}&\Delta_{\text{adapt}}(Hw, LM, B) \\ &= P_{\text{S}}(Hw)\left|\log P_{\text{S}}(w|H) - \log P_{\text{LM+B}}(w|H)\right|\end{aligned}$$

where $P_{\text{LM+B}}$ denotes the probability of an n-gram in the biased LM using all previously selected n-grams.

Given a threshold t, we select Hw if

$$\Delta_{\text{adapt}}(Hw, LM, B) > t \tag{1}$$

Algorithm 1 describes this process:

Algorithm 1 n-gram selection by threshold.

```
    Given a probability distribution of $n$-grams in the sample $S$, a general language model $LM$, a divergence threshold $t$, and a maximum $n$-gram length $n$, returns a set $B$ of $n$-grams and costs, such that each $n$-gram added of length $k$ accounts for $t$ or more of $D_{KL}$ between $S$ and $(LM + B^{k-1})$ ($B^i$ being the set of selected ngrams of length up to $i$).
1:  function SelectNgrams($S, LM, t, n$)
2:      $B \leftarrow \emptyset$                              ▷ The biasing model
3:      for $i = 1 \ldots n$ do                  ▷ from unigrams up to length $n$
4:          for $Hw \in S^i$ do                        ▷ $n$-grams of length $i$
5:              if $|\text{Cost}(LM, B, Hw) + \log P_{\text{S}}(w|H)| > t$ then
6:                  $B \leftarrow B \cup (Hw, -\log P_{\text{S}}(w|H))$
7:              end if
8:          end for
9:      end for
10:     return $B$
11: end function

    Returns the biased $n$-gram cost of $Hw$, defined as the cost of the longest non-empty suffix $H'w \in B$ or the $LM$ cost of $Hw$, if $B$ contains no such $H'w$.
12: function Cost($LM, B, Hw$)
13:     $n \leftarrow |H|$
14:     for $i = n - 1 \ldots 0$ do
15:         $H' \leftarrow \text{IthSuffix}(H, i)$
16:         if $H'w \in B$ then
17:             return $B[H'w]$        ▷ Return the biasing model cost
18:         end if
19:     end for
20:     return $-\log P_{\text{LM}}(w|H)$             ▷ Return the LM cost
21: end function
```

3. MODEL SIZE OPTIMIZATION

In the previous section, we describe the n-gram selection process given a threshold t. In this section, we describe how the value for t is chosen. We would like to find a t such that the model captures a significant part of D_{KL} while remaining small. The challenge in finding the optimal t is that it depends on the context. Section 4.4 expands on this and other differences between contexts.

Our method first estimates the context's total D_{KL} compared to the LM. Given this estimate, we then set t such that the final model covers $p\%$ of the D_{KL} estimate.

To evaluate the biasing model's D_{KL}, we define the improvement expected from adding a new n-gram Hw to the biasing model B, Δ_{KL}:

$$\begin{aligned}&\Delta_{\text{KL}}(B, Hw) = \\ &\quad P_{\text{S}}(Hw)(\,|\log P_{\text{S}}(w|H) - \log P_{\text{LM}}(w|H)| - \\ &\quad\quad |\log P_{\text{S}}(w|H') - \log P_{\text{LM}}(w|H')|)\end{aligned}$$

Where $H'w$ is the longest n-gram suffix of Hw that is already included in the biasing model B.

Summing Δ_{KL} for $H'w$ and Hw, we get (assuming the model consists of these two n-grams):

$$\begin{aligned}&\Delta_{\text{KL}}(B, H'w) + \Delta_{\text{KL}}(B, Hw) = \\ &\quad (P_{\text{S}}(H'w) - P_{\text{S}}(Hw))\,|\log P_{\text{S}}(w|H') - \log P_{\text{LM}}(w|H')| \\ &\quad + P_{\text{S}}(Hw)\,|\log P_{\text{S}}(w|H) - \log P_{\text{LM}}(w|H)|\end{aligned}$$

$(P_{\text{S}}(H'w) - P_{\text{S}}(Hw))\,|\log(P_{\text{S}}(w|H')) - \log(P_{\text{LM}}(w|H'))|$ accounts for the contribution we expect to get from adjusting the n-gram $H'w$: We expect this adjustment to take effect when scoring an n-gram ending with $H'w$, excluding n-grams ending with Hw (for those we use Hw). $P_{\text{S}}(Hw)\,|\log P_{\text{S}}(w|H) - \log P_{\text{LM}}(w|H)|$ accounts for the contribution from adjusting the n-gram Hw.

When summing $\Delta_{\text{KL}}(B, Hw)$ for all n-grams in B, we thus get an estimate for the total reduction in KL-divergence contributed by B:

$$\Sigma\Delta_{\text{KL}}(B) = \sum_{Hw \in B} \Delta_{\text{KL}}(B, Hw) \tag{2}$$

We denote B^* as the biasing model containing all the n-grams in the sample. Similarly, we denote

$$\Sigma\Delta_{\text{KL}} = \Sigma\Delta_{\text{KL}}(B^*) \tag{3}$$

$\Sigma\Delta_{\text{KL}}$ gives us an estimate of the total possible reduction in KL-divergence that we would have if we would select all the n-grams from the sample.

Computing $\Delta_{\text{KL}}(B, Hw)$ for each n-gram $Hw \in B$ can be done as part of algorithm 1, and so our approach is the following:

1. Run algorithm 1 with $t = 0$. This will select all n-grams into B. For each n-gram Hw, also calculate $\Delta_{\text{KL}}(B, Hw)$. Because all the n-grams are selected, $\Sigma\Delta_{\text{KL}}(B) = \Sigma\Delta_{\text{KL}}$. (See equation 3)
2. Sort n-grams by Δ_{adapt}.
3. Starting from the highest Δ_{adapt} going down, sum each n-gram's $\Delta_{\text{KL}}(B, Hw)$, until the sum is greater than $p\%$ of $\Sigma\Delta_{\text{KL}}$ calculated in step 1.
4. Set $t = \Delta_{\text{adapt}}$ of the first n-gram not counted.
5. Run algorithm 1 with the newly calculated t.

Note that the algorithm described is an approximation. $\Delta_{\text{KL}}(B, Hw)$ depends on the n-grams previously selected $H'w$. An n-gram's $\Delta_{\text{KL}}(B, Hw)$ summed in step 2 may be different from its $\Delta_{\text{KL}}(B^*, Hw)$ in the final model (step 5), because the model may then contain a different $H'w$. Recall also that if the model contains an n-gram Hw, that n-gram may override the LM score for n-grams with longer history, while our definition of $\Delta_{\text{KL}}(B, Hw)$ assumes that the LM score replaced is always $-\log(P_{\text{LM}}(Hw))$. That being said, these approximations work very well, so there is no practical need for a more complex (and slower) algorithm.

4. EXPERIMENTAL RESULTS

In this section we describe our experimental setup and context types used, and analyze results. To evaluate our models, we ran two types of experiment:

1. Offline experiments on human-transcribed test sets.
2. Live traffic side-by-side experiments rated by humans.

All the utterances used (in test sets or side-by-side experiments) have been anonymized. The test sets described below all consist of human-transcribed utterances in American English. All the biasing models described in this paper were trained using unsupervised voice queries from live traffic.

Our baseline system consists of a long short-term memory acoustic model [11, 12] and a Katz smoothed [13] 5-gram finite state transducer [14, 15] LM, pruned to 100M n-grams and trained using Bayesian interpolation [6]. The system also includes a larger second pass LM, trained on the same data [16]

4.1. Biasing model training

- All our models were trained using a minimum n-gram length of 2. Eliminating unigrams helped reduce over-triggering as the models were able to capture enough most of the contextual information using bigrams and higher order n-grams.
- When setting the cost of each n-gram in the model, we added a penalty of 2 on top of the cost derived from the sample probability:

  $$Cost(wH) = -\log(P_{\text{S}}(w|H)) + 2$$

  The penalty serves to slightly weaken the model's biasing. This value is optimized on a develepment testset so that it compensates for the fact that the biasing models are trained on far less data than the main LM.

4.2. Confirmation Context Type

The Confirmation context corresponds to a dialog state in which the user is expected to confirm or cancel some action. The main test set, "CONFIRM", consists of 976 human-transcribed, randomly sampled utterances. Typically, these utterances contain n-grams such as {"yes", "no", "yeah", "send it", "change it"}. An additional test set, "GENERAL", consists of $13K$ utterances that do not correspond to the confirmation dialog state. This test set is meant for evaluating biasing over-triggering.

The Confirmation biasing models were trained using $1.76M$ unsupervised, randomly sampled utterances. We evaluated 3 unsupervised learning models, containing 90%, 95% and 100% of $\Sigma\Delta_{\text{KL}}$. These models were compared against a baseline (no biasing) and against a manual model. The manual model consists of n-grams built from 619 manually chosen relevant phrases, using the process described in [3]. Table 1 shows the size of the models evaluated.

manual	90%	95%	100%
3475	60	289	9635

Table 1. Number of n-grams in the Confirmation biasing models: The manual model (built from 619 manually chosen relevant phrases, broken into n-grams [3]) and the unsupervised-learning models with 90%, 95% and 100% of $\Sigma\Delta_{\text{KL}}$.

The results in table 2 show that the unsupervised-learning models perform better than both the baseline and the manual-phrase model in GENERAL & CONFIRM test sets.

Test set	baseline	manual	90%	95%	100%
CONFIRM	19.1	13.6	11.8	11.8	12.1
GENERAL	12.0	12.3	12.0	12.0	12.1

Table 2. WER(%) for the baseline, the manual model, and the unsupervised-learning models with 90%, 95% and 100% of $\Sigma\Delta_{\text{KL}}$.

In particular, the Confirmation unsupervised-learning models with 90% and 95% of $\Sigma\Delta_{\text{KL}}$ produced the best results, without having a negative effect on the GENERAL set.

As the number of biasing n-grams increases above 90% of $\Sigma\Delta_{\text{KL}}$, no more reduction in WER is observed, and an increase in over-triggering causes a slight increase in the GENERAL test set WER.

4.3. Side-by-side experiments

We also ran "side-by-side" (SxS) experiments on live traffic. In these experiments each utterance is automatically transcribed by two systems: baseline and experimental. If the two transcripts are not identical, they are sent for rating. The experiment continues until there 500 such utterances. Each utterance is rated by two humans (and a third in case of a tie). For each of the SxS experiments, we present the following:

% Changed The percentage of utterances in which the two systems produced different transcripts.

Wins/Losses The ratio of wins to losses in the experimental system vs. the baseline.

4.3.1. Confirmation Context Type

The results for the Confirmation models described in 4.2 are below: Table 3 compares the best unsupervised-learning models to a baseline with no context. Table 4 compares all three unsupervised models to a baseline with the manual biasing model.

Exp	% Changed	Wins/Losses
manual	6.81	1.84
90%	1.33	7.23

Table 3. SxS results of the Confirmation manual-phrase model and the unsupervised-learning model with 90% of $\Sigma\Delta_{\text{KL}}$, vs. a baseline with no contextual biasing.

Exp	% Changed	Wins/Losses
90%	5.67	2.24
95%	5.7	1.83
100%	5.94	1.33

Table 4. SxS results of Confirmation unsupervised-learning models with 90%, 95%, 100% of $\Sigma\Delta_{\text{KL}}$, vs. the manual-phrase model.

4.3.2. Text message dialog states

These dialog states occur in the process of the user sending a text message. For each such state, we used a biasing model set to keep 95% of its sample's $\Sigma\Delta_{\text{KL}}$. Table 5 compares each model vs. a baseline with no contextual biasing.

text_confirm: The user is asked if the message should be sent. This model has 156 n-grams, such as: "$\langle S \rangle$ send", "send it"and "change it".

recipient: The user is asked who is the recipient of the message. The model has 1574 n-grams, such as: "$\langle S \rangle$ \$CONTACTS", "$\langle S \rangle$ \$FULLPHONENUM" and "$\langle S \rangle$ text \$CONTACTS".

type: Contact type disambiguation. The user is asked to choose one of several contact entries for the intended recipient. The model has 26 n-grams, such as: "$\langle S \rangle$ mobile", "first one", "$\langle S \rangle$ work" and "$\langle S \rangle$ cancel".

name: Contact name disambiguation. The user is asked to choose one of several contacts with similar names. The model has 1492 n-grams such as: "$\langle S \rangle$ mom", "text \$CONTACTS" and "another contact".

Exp	% Changed	Wins/Losses
text_confirm	14.85	26.73
recipient	2.81	2.42
type	2.82	3.76
name	5.70	3.33

Table 5. SxS results of Text Message unsupervised-learning models compared to a baseline with no contextual biasing.

Note the "text_confirm" state experiment, showing significant gains using only 156 biasing n-grams. Some examples of biasing wins are: "egg" $\rightarrow$ "add", "council" $\rightarrow$ "cancel" and "Joan Jett" $\rightarrow$ "change it".

4.3.3. Application specific biasing

In these experiments, the models were trained using speech queries spoken within the YouTube application. The experiments shown here used queries spoken in Russian (ru-ru) and Brazilian Portuguese (pt-br). For each locale, we evaluated models with $1K$ and $10K$ n-grams. Table 6 shows the results of each of these models vs. a baseline with no context.

Exp	% Changed	Wins/Losses
ru-ru_1K	3.28	2.03
ru-ru_10K	6.18	2.65
pt-br_1K	1.98	1.44
pt-br_10K	4.25	1.26

Table 6. SxS results of YouTube unsupervised-learning models vs. a baseline system with no contextual biasing.

The results in table 6 show that the larger models ($10k$) performed better than the smaller ones($1k$) (pt-br_$10K$ has a slightly lower Wins/Losses ratio than pt-br_$1K$, but a much higher %Change). This is unlike the Confirmation setting in section 4.2, where the best model had a mere 60 n-grams. In the following section we demonstrate the difference between these two contexts.

4.4. Comparing contexts

The results shown in sections 4.2 and 4.3.3 show that The optimal model size varies between contexts. In this section, we expand on this comparison.

Figure 1 shows that in the YouTube context, far more n-grams are required to reach a substantial $\Sigma\Delta_{\text{KL}}(B)$ compared to the Confirmation context. The figure shows that with 60 n-grams, the Confirmation model reaches 90% of its sample's $\Sigma\Delta_{\mathrm{KL}}$. With 60 n-grams, the YouTube model reaches less than 30% of its sample's $\Sigma\Delta_{\mathrm{KL}}$.

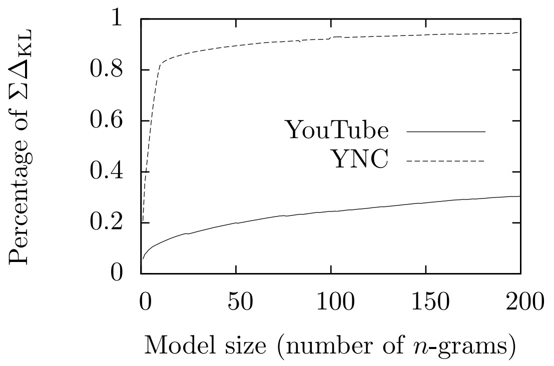

Fig. 1. Model size and percentage of $\Sigma\Delta_{\mathrm{KL}}$.

In section 3, we stated that there isn't one optimal value for the threshold t used in the selection process. In figure 2, we see that setting $t = 0.002$ would result in over 90% of the $\Sigma\Delta_{\mathrm{KL}}$ of the Confirmation model, but in only 20% of the $\Sigma\Delta_{\mathrm{KL}}$ of the YouTube context. In the YouTube context, to get 80% of the $\Sigma\Delta_{\mathrm{KL}}$ we would need to set $t = 0.00005$.

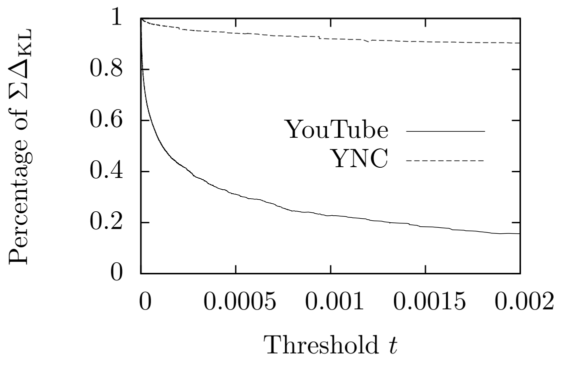

Fig. 2. Value of t and the percentage of $\Sigma\Delta_{\mathrm{KL}}$.

Finally, figure 3 shows another important difference between the two contexts. While $\Sigma\Delta_{\mathrm{KL}}$ estimated for the Confirmation sample is over 2.5, $\Sigma\Delta_{\mathrm{KL}}$ for the YouTube sample is less than 1.

This comparison demonstrates why percentage of $\Sigma\Delta_{\mathrm{KL}}$ is useful for choosing the biasing model's size—$\Sigma\Delta_{\mathrm{KL}}$ naturally adapts to the given context, whereas other natural candidates (constant value of t, hard limit on the number of n-grams or on $\Sigma\Delta_{\mathrm{KL}}(B)$) work well in some contexts but not in others.

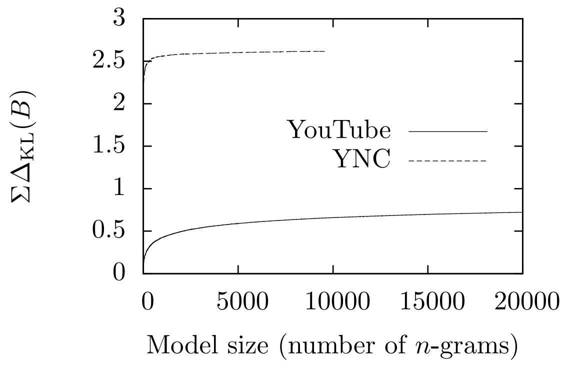

Fig. 3. Model size and $\Sigma\Delta_{\mathrm{KL}}(B)$.

5. CONCLUSIONS

In this paper we presented a system that uses unsupervised data to learn contextual information. Our system can exploit various types of context. It can be used to automatically build a large number of contextual models from a small amount of unsupervised training data. We also described how to automatically select n-grams relevant for a particular context as well as how to choose an optimal size for contextual models.

We presented how such models can be used to increase ASR accuracy in particular contexts. The experimental results show large improvements in recognition accuracy when contextual models are used for all the contexts that we analyzed. We also show large improvements in metrics obtained in human rated experiments.

Acknowledgements

Thanks to Pedro Moreno and Clara Gordon for their help.

6. REFERENCES

[1] Jerome R. Bellegarda, “Statistical language model adaptation: review and perspectives,” *Speech Communication*, vol. 42, pp. 93–108, 2004.

[2] Reinhard Kneser and Volker Steinbiss, “On the dynamic adaptation of stochastic language models,” in *Proceedings of ICASSP*, 1993, pp. 586–589.

[3] Petar Aleksic, Mohammadreza Ghodsi, Assaf Michaely, Cyril Allauzen, Keith Hall, Brian Roark, David Rybach, and Pedro Moreno, “Bringing contextual information to google speech recognition,” in *Interspeech 2015*, 2015.

[4] Keith B. Hall, Eunjoon Cho, Cyril Allauzen, Francoise Beaufays, Noah Coccaro, Kaisuke Nakajima, Michael Riley, Brian Roark, David Rybach, and Linda Zhang, “Composition-based on-the-fly rescoring for salient n-gram biasing,” in *Interspeech 2015*, 2015.

[5] Brandon Ballinger, Cyril Allauzen, Alexander Gruenstein, and Johan Schalkwyk, “On-demand language model interpolation for mobile speech input,” in *Interspeech*, 2010, pp. 1812–1815.

[6] Cyril Allauzen and Michael Riley, “Bayesian language model interpolation for mobile speech input,” in *INTERSPEECH*, 2011, pp. 1429–1432.

[7] Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai, “Class-based n-gram models of natural language,” *Computational Linguistics*, vol. 18, pp. 467–479, 1992.

[8] Lucy Vasserman, Vlad Schogol, and Keith B. Hall, “Sequence-based class tagging for robust transcription in asr,” in *Proc. Interspeech*. September 2015, ISCA - International Speech Communication Association.

[9] Petar Aleksic, Cyril Allauzen, David Elson, Aleksandar Kracun, Diego Melendo Casado, and Pedro J. Moreno, “Improved recognition of contact names in voice commands,” in *ICASSP*, 2015, pp. 5172–5175.

[10] Solomon Kullback, “Letter to the editor: The kullback-leibler distance,” 1987.

[11] Haşim Sak, Andrew Senior, and Françoise Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” in *Interspeech*, 2014.

[12] Haşim Sak, Oriol Vinyals, Georg Heigold, Andrew Senior, Erik McDermott, Rajat Monga, and Mark Mao, “Sequence discriminative distributed training of long short-term memory recurrent neural networks,” in *Interspeech*, 2014.

[13] Slava M. Katz, “Estimation of probabilities from sparse data for the language model component of a speech recognizer,” in *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1987, pp. 400–401.

[14] Mehryar Mohri, Fernando Pereira, and Michael Riley, “Weighted finite-state transducers in speech recognition,” *Computer Speech and Language*, vol. 16, pp. 69–88, 2002.

[15] Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri, “OpenFst: A general and efficient weighted finite-state transducer library,” in *CIAA 2007*, 2007, vol. 4783 of *LNCS*, pp. 11–23, `http://www.openfst.org`.

[16] Preethi Jyothi, Leif Johnson, Ciprian Chelba, and Brian Strope, “Distributed discriminative language models for google voice search,” in *Proceedings of ICASSP 2012*, 2012, pp. 5017–5021.