ENTROPY BASED PRUNING OF BACKOFF MAXENT LANGUAGE MODELS WITH CONTEXTUAL FEATURES

Tongzhou Chen [*1] Diamantino Caseiro [†] Pat Rondon [†]

[*]Georgia Institute of Technology, [†]Google, Inc.

Introduction

We present a pruning technique for maximum entropy (MaxEnt) language models. It is based on computing the exact entropy loss when removing each feature from the model, and it explicitly supports backoff features by replacing each removed feature with its backoff. The algorithm computes the loss on the training data, so it is not restricted to models with n-gram like features, allowing models with any feature, including long range skips, triggers, and contextual features such as device location.

Background

A maximum entropy language model with backoff-inspired features consists of:

- A set X of word contexts such as previous history or contextual information.
- A set Y denotes the vocabulary.
- A set of functions $f_i : X \times Y \mapsto \mathbb{R}^d$ mapping any (x, y) pair to a sparse $0-1$ feature vector in $\mathbb{R}^d$. $f(x,y) = 1$ if some property of x with y is true; and 0 otherwise.
- A parameter vector v in $\mathbb{R}^d$.
- Optionally, a set of functions $bo_i : X \times Y \mapsto \mathbb{R}$ assigning backoff weights. $bo_i(x,y) \neq 0$ only when the corresponding $f_i(x,y)$ does not exist.

For any $x \in X$ and $y \in Y$, a MaxEnt model gives the posterior as:

$$p(y|x;v) = \frac{N(y|x;v)}{Z(x;v)},$$

where

$$N(y|x;v) = \exp(\sum_i v \cdot f_i(x,y) + bo_i(x,y)), \quad Z(x;v) = \sum_{y' \in Y} N(y|x;v).$$

Entropy Based Pruning

Goal: To minimize the relative entropy between the original and pruned models.

Equivalently, it is to maximize the difference of the log-likehoods between the pruned and original models.

Loss $L_{-f_i(x,\tilde{y})}$: log-likelihood difference of model without feature $f_i(x,\tilde{y})$ and the original model, in the training data $D = \{(x,y)\}$,

$$L_{-f_i(x,\tilde{y})} = \sum_{(x,y)\in D} \log P_{-f_i(x,\tilde{y})}(y|x;v) - \log P(y|x;v).$$

where

$$p_{-f_i(x,\tilde{y})}(y|x;v) = \frac{N_{-f_i(x,\tilde{y})}(y|x;v)}{Z_{-f_i(x,\tilde{y})}(x;v)},$$

$$N_{-f_i(x,\tilde{y})}(y|x;v) = \begin{cases} \exp(\log N(y|x;v) - v_i + bo(x,\tilde{y})) & \text{if } y = \tilde{y}; \\ N(y|x;v) & \text{otherwise,} \end{cases}$$

$$Z_{-f_i(x,\tilde{y})}(x;v) = Z(x;v) - N(y|x;v) + N_{-f_i(x,\tilde{y})}(y|x;v).$$

The term $bo(x,\tilde{y})$ is removed when backoff features are not used in the model.

The complexity of computing loss for all features is the same as one epoch of inference in training.

Algorithm

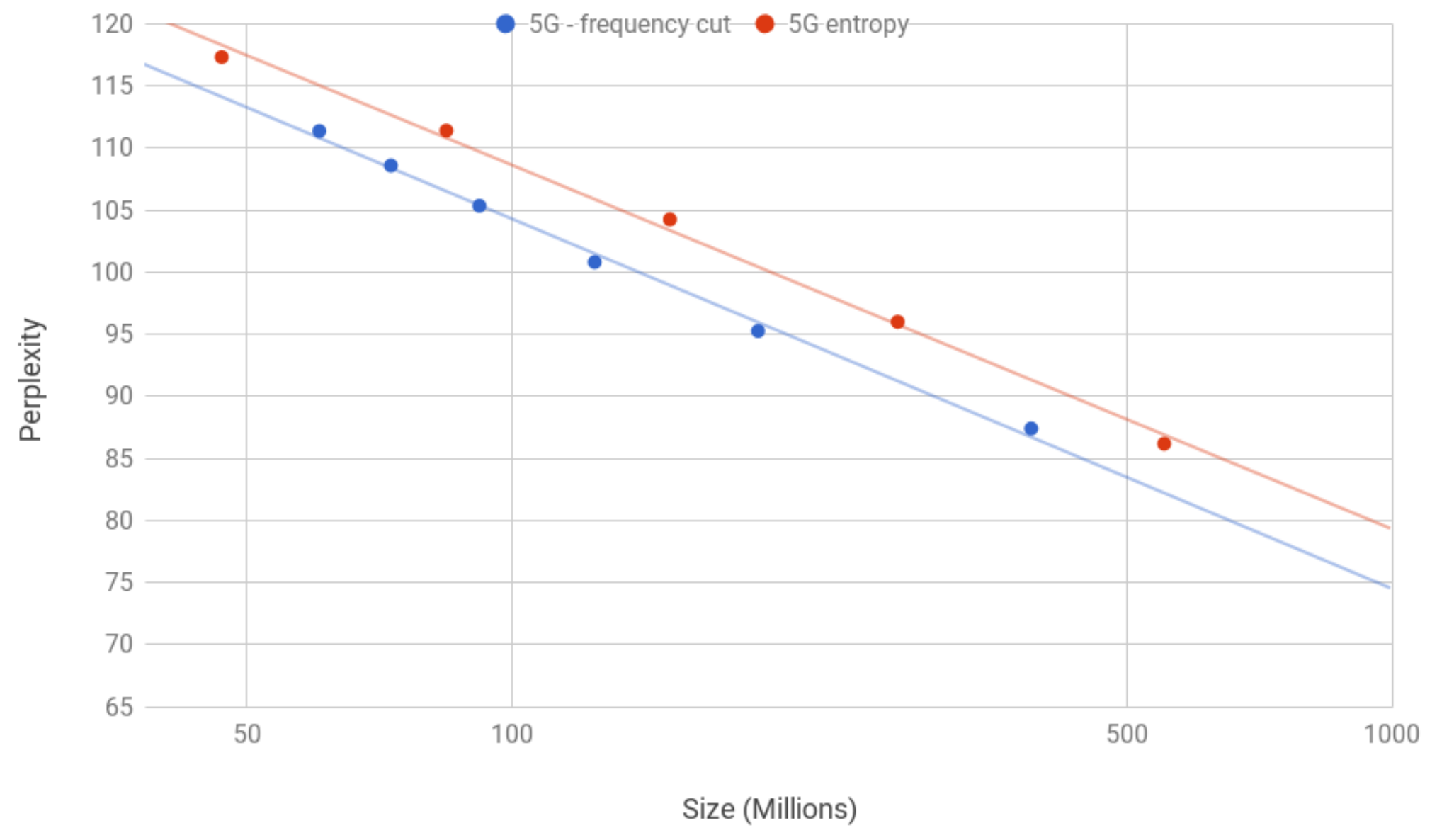
- We first estimate the entropy loss per feature on a trained model.
- Then we prune the features with loss no less than some threshold.
- Finally, we retrain the pruned model with weights starting from 0 and evaluated the PPL and WER on the pruned, retrained model.

Remark: Unigram features and backoff features are never pruned.

All our models are hierarchical $P(w|h) = P(c(w)|h)P(w|x,h)$ and use 1000 clusters.

Overcoming Overfiting

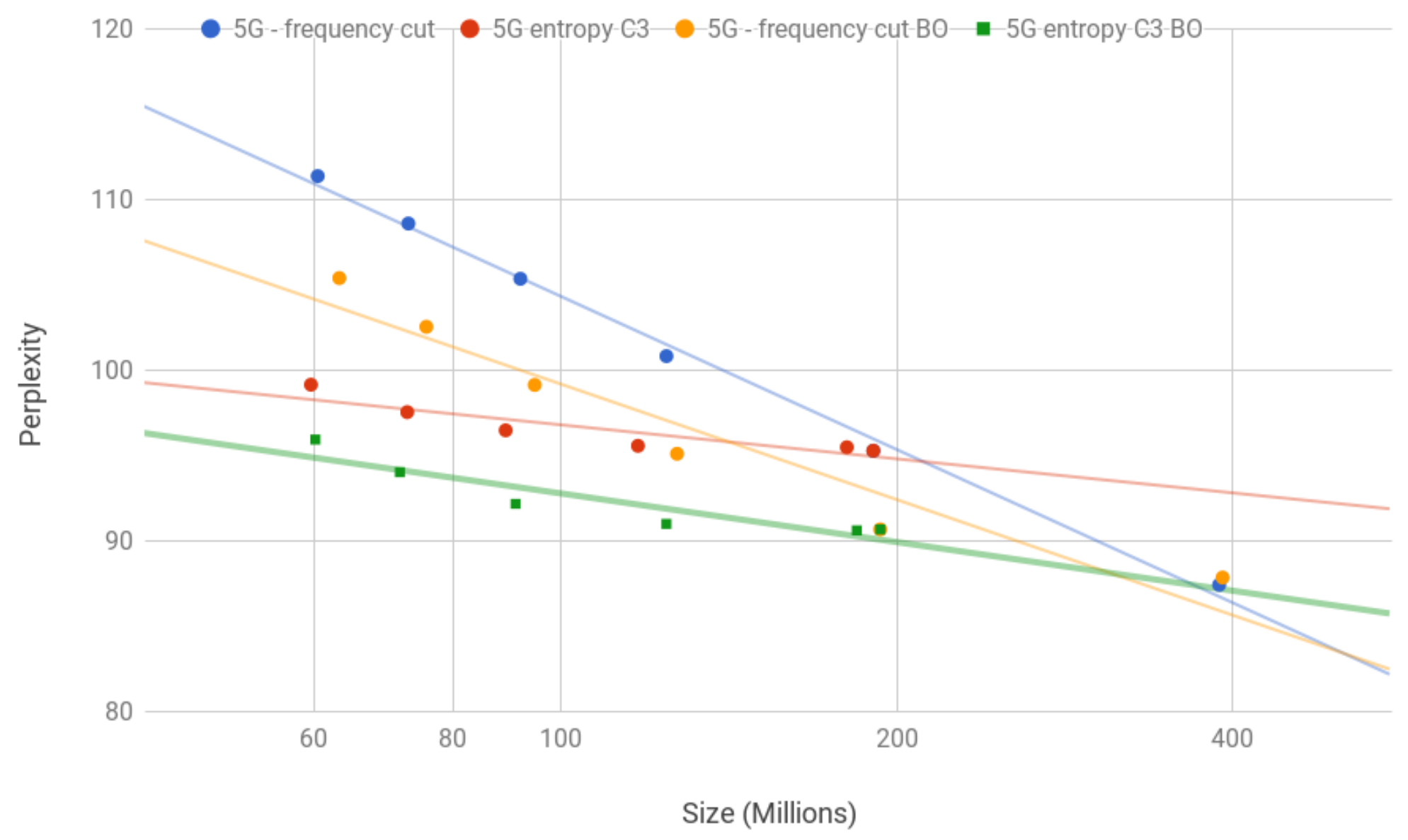
The first experiments was done on the one billion corpus with n-gram features only (up to 5-gram). Compared to frequency pruning, entropy pruning perplexity is about 5 points worse across a wide range of model sizes on n-gram model.



Looking at the number of features selected by template we notice that entropy pruning is selecting mostly singleton features, thus overfiting to the training data.

	2-gram	3-gram	4-gram	5-gram
% kept	9.09%	7.64%	19.67%	39.75%

To overcome this problem, we decided to filter out features with low frequencies before applying entropy pruning. Entropy pruning achieves much better perplexity on n-gram model with features occuring at least 3 times in the training data. The proposed entropy pruning methods is also effective for models with backoff features.



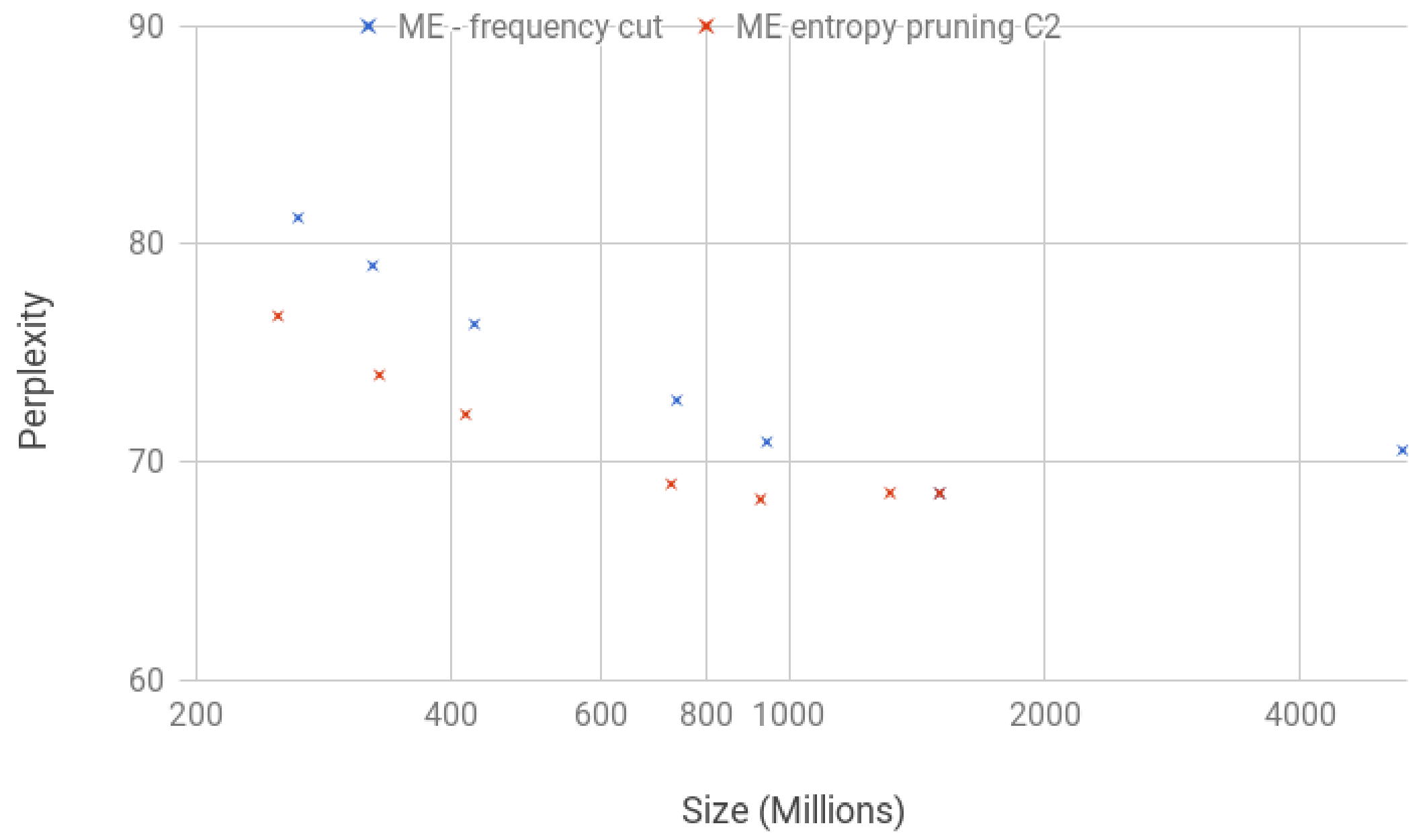
Analysing the behaviour of entropy pruning by feature template, we now see that the pruning algorithm prefers to keep more general features such as 2-grams and 3-grams.

Threshold	2-gram	3-gram	4-gram	5-gram
0	93.71%	94.53%	94.63%	95.35%
-0.3	69.89%	63.05%	59.94%	58.91%
-0.8	55.03%	47.29%	44.82%	45.77%
-1.3	46.93%	38.49%	35.76%	37.70%
-1.9	40.77%	31.71%	28.46%	30.94%

Beyond n-grams

To verify if the proposed entropy pruning model is also effective in models with a larger variety of features types, we trained a MaxEnt model on the one billion corpus with word n-grams, word cluster n-grams, skip 2-grams, left and right skip 3-grams.

To avoid overfiting of the pruning algorithm, we did not include singleton features in the initial model. Entropy pruning gives significant improvements in perplexity.



Geographic Adaptation Experiments

The following experiment was done on a model for US English with geographic adaptations. We performed city-specific prunings which keep the features *within each city* whose losses are in the 95-th percentile of the features in that city; we used the largest three US cities for our experiment.

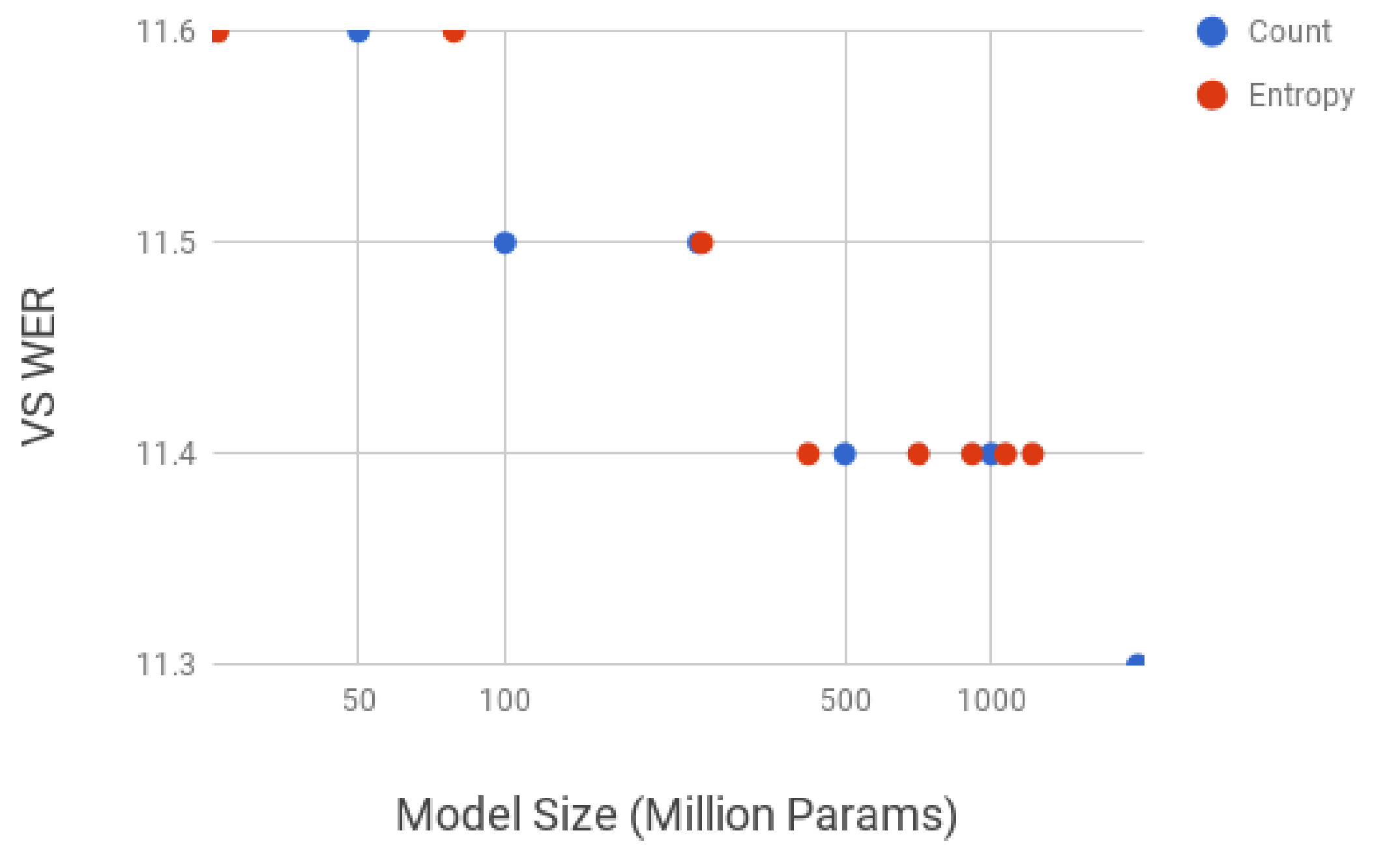
Among the top five unigrams by loss in each city, there are unigrams with clear city-specific importance.

New York	Los Angeles	Chicago
kcbs	los	what
brooklyn	what	how
staten	how	what's
bronx	burbank	chicago
manhattan	northridge	cubs

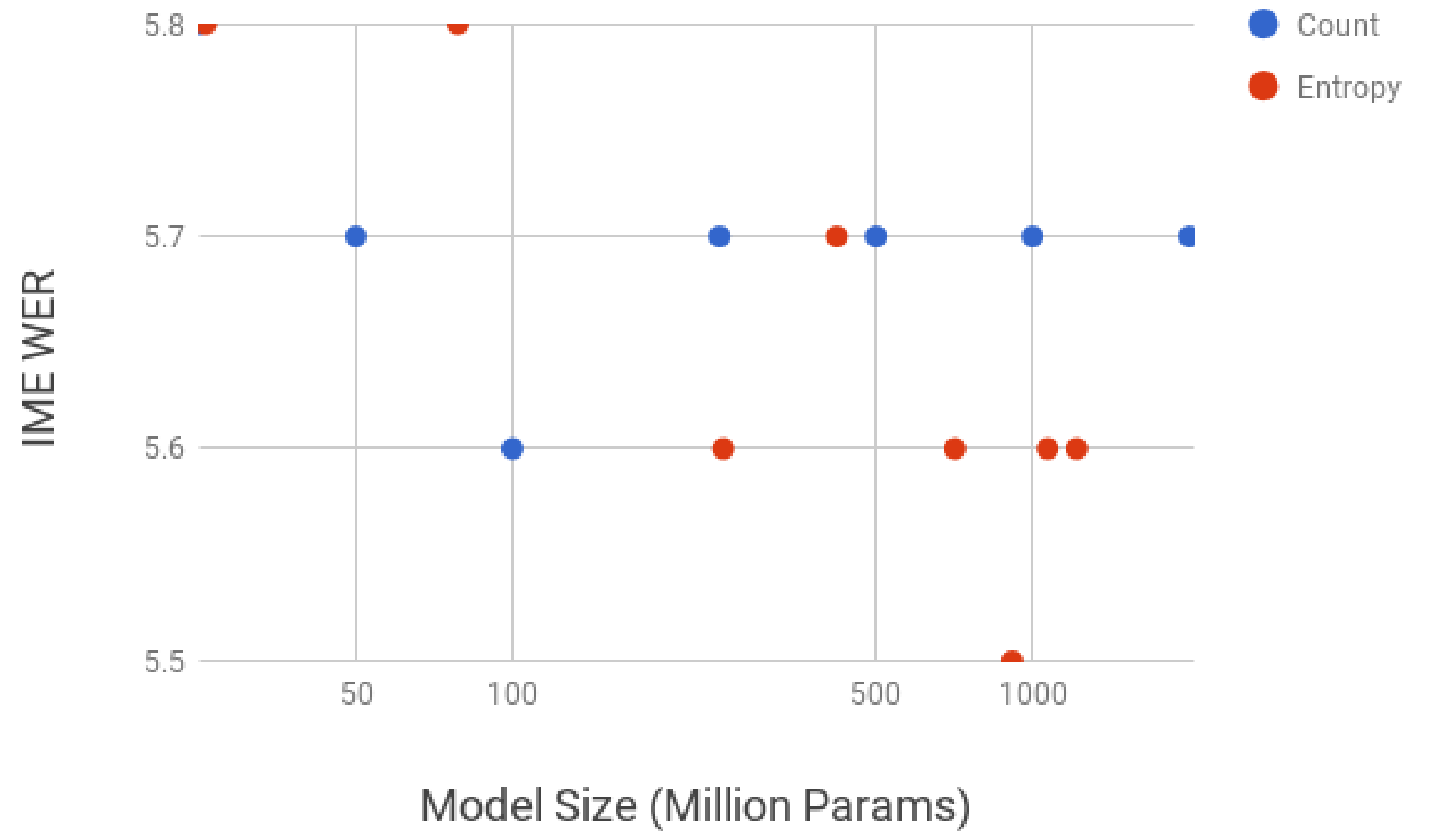
ASR Experiments

We conducted automatic speech recognition (ASR) experiments to the impact of model pruning on automatic speech recognition quality. All experiments were based on Google's cloud based mobile ASR system for Italian.

In the voice search test set, WER using entropy pruning is basically the same as using frequency pruning.



In the dictation test set entropy pruned models achieve some improvements in WER.



Conclusion

We showed that the proposed pruning algorithm for MaxEnt models leads to significantly better models, in terms of perplexity and WER, than frequency pruning.

Results on the 1-billion word corpus show large perplexity improvements relative for frequency pruned models of comparable size.

Automatic speech recognition (ASR) experiments show WER improvements in a large-scale cloud based mobile ASR system for Italian.

[a] This author performed the work while at Google, Inc.