2011 International Conference on Document Analysis and Recognition

Limits on the Application of Frequency-based Language Models to OCR

Ray Smith
Google Inc
Mountain View, USA
rays@google.com

***Abstract*—Although large language models are used in speech recognition and machine translation applications, OCR systems are "far behind" in their use of language models. The reason for this is not the laggardness of the OCR community, but the fact that, at high accuracies, a frequency-based language model can do more damage than good, unless carefully applied. This paper presents an analysis of this discrepancy with the help of the Google Books n-gram Corpus, and concludes that noisy-channel models that closely model the underlying classifier and segmentation errors are required.**

Keywords-OCR; language models.

I. Introduction

Language models are essential in OCR applications. In English, the I/l/1 and O/0 ambiguities alone prove the point, so OCR systems have had both word-lists and character n-grams for two decades[1]. Yet the most common question addressed to the author over more than two decades in OCR is: "Why don't you use a dictionary?" Users continue to see "stupid" errors, where an obvious dictionary word is mis-recognized. If a spelling checker can find the correct word, surely a better language model correctly integrated with the shape classifier would be able to do much better?

The answer is that the weight of the language model is optimized for average accuracy, and that to increase its weight would fix some of the obvious errors, but would increase the rate of hallucination of incorrect dictionary words. While this ought to be true for *any* optimal system, it says nothing about the effect of increasing the sophistication of the language model, and with that the selectivity of its corrections, which should improve the optimal accuracy.

Speech recognition systems have been using more sophisticated language models than OCR systems for decades[2][3], but OCR systems have not followed. There are several possible reasons for this dichotomy of methods:

- With roots in the 1980s, software OCR applications have a small memory footprint, and find it hard to justify the factor of 100 or more increase in memory required to add larger language models.
- OCR researchers have tried larger language models, but found them to be unsuccessful. Since negative results are rarely reported, this has remained a well-kept "secret" of the OCR community.
- OCR researchers are laggards, and have not put in the effort necessary to get the most out of better language models.

The success of speech systems with language models[4], and the BBN Byblos system[5] with Arabic OCR makes the second reason above "impossible." In this paper, we set out to investigate the factors that affect the optimal accuracy in an OCR system that uses a language model, and how it might be possible that language models are less useful to OCR than to speech recognition systems. The investigation applies a simplified model of an OCR shape classifier and different language models (defined in Section III) to the large Google Books n-gram Corpus[6] of 10^{11} words.

II. Background and Zipf's Law

Zipf's law[7] states that when the words in a language are ranked by decreasing frequency, the word with rank n has a frequency proportional to $1/n$. Consequently the 100 most frequent words in English account for more than a third of all words in a corpus. Frequent words are so consistent that they form the foundation of attacks on simple ciphers[8], and similar methods have been applied more than once[9][10] to OCR text with no initial classifier training.

The other end of the $1/n$ distribution is the "long tail" and indicates that a large language model is never complete, however large it may be, so there will always be words or phrases that fool a language model by being improbable.

For example, consider 'arıd' which could be 'and' with the 'n' broken, or 'arid' with the dot missing from the 'i'. Since 'and' is much more frequent than 'arid,' the language model would need decisive context to be able to accept 'arid.' Yet in: 'Capability Brown terms arıd conditions ...,' most humans and language models, would still prefer 'and' without either the continuation '... as annual rainfall of less than ...' or the knowledge that Capability Brown was an 18th Century English landscape gardener.

The previous example is concocted, but a reality is even more bizarre. At one point in its development, Tesseract[11] recognized '8½"' as 'oven' simply by extracting the top-choice dictionary word from the n-best lists of the character classifier, with the error mapping 8->o, 1/->v, 2->e, "->n.

Anecdotal examples are interesting, but not convincing. The data set used in section IV, is the the Google Books English n-gram corpus[6], which consists of roughly 4×10^6 unique words in a collection of 10^{11} total words. The public corpus has been filtered[13] to include only words that occur in at least 40 books, but the unfiltered version contains more than 10^8 unique words. We split the corpus into a Dictionary D and a test Corpus C. $W \in C$ is a truth word of m symbols or shapes and $\hat{W} \in D$ is a candidate result word.

III. Methodology

To provide a reasonably simple analysis, in this paper we consider the case of an isolated shape classifier, combined with a language model that has a word n-gram frequency

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.114

model ($1 \le n \le 3$) or a binary n-gram dictionary model. In each case, the shape classifier offers a list of shapes, each with a "probability." HMM-based models that allow a shape to consist of multiple model states, such as the BBN system[5] are not considered. Choices in the simplifying assumptions have been made on the side of over-stating the final accuracy, and thus providing an upper bound on the improvements that can be made with a language model.

A. Classifier Model

For each shape classified, (a shape may be anything from a glyph to a grapheme cluster) the shape classifier returns a top choice shape s_{tc} with probability $p(s_{tc})$ and also **all** other shapes in the shape set $s_i : 0 \le i < N_s,\ i \ne tc$ with probability $p(s_i)$, with N_s the size of the shape set. For simplicity, $p(s_i)$ is equal for all $i \ne tc$ i.e. all shapes other than the top choice have equal probability. Furthermore, this relationship holds whether s_{tc} is correct or not, i.e. the classifier is equally as confident in its errors than its correct choices.

This classifier model is a gross simplification. In any real OCR system, one would expect the shape classifier to occasionally return multiple shapes with almost equal, or even exactly equal probabilities, for example for I/l/1. One would also hope that the correct answer, if not the top choice, would at least be near the top choice. While often true, it is also true that an incorrect choice is often near the correct choice, (eg. with I/l/1) and it is also often the case that the correct answer is nowhere to be found in the n-best list provided by the shape classifier. The only chance the language model has to recover from such catastrophic errors is to allow wildcards. The simplified model used here essentially allows the language model to use wildcards in every case, and weights control how readily they are applied.

To simplify further, we assume that the shapes to be recognized are perfectly segmented. We therefore call the concatenation of the top-choice classifier shapes $s_{TC,j}$ at each position $j \in [1, m]$ the Top-choice Classifier Word, W_{TC}. Although also a gross simplification, and segmentation is necessary in a real OCR system, the error model does not include any segmentation errors, so segmentation errors do not need to be corrected, and adding them would only increase the scope for the language model to hallucinate incorrect dictionary words.

Another simplifying assumption is that shape classifier errors are statistically independent. In reality errors often occur in bursts due to image quality problems, but the assumption of statistical independence allows the following simple analysis of the shape vs. word error rates: if the probability of a top-choice shape classifier error is p_e, and errors are statistically independent, then the probability of a single top-choice error in W of length m is $mp_e(1-p_e)^{m-1}$ and the total probability of error is $1-(1-p_e)^m$.

B. Word n-gram Language Model

The language model chooses the best path through the segmentation graph that optimizes the combined probability, using the log-linear model[12] to weight the language model against the shape classifier. For $n>1$ a beam search would normally be used to choose the overall few-best sequence of words, but since C is a collection of n-grams instead of flowing text, the treatment for $n>1$ is simplified to hold n-1 of the words constant (the context) and consider only a single word for error simulation/correction. The context thus partitions C and D into many separate cases in combination with the word length m.

Since we have no segmentation errors, The language model chooses the best path through the lattice such that:

$$cost = -w_{LM} \ln p(\hat{W}) - w_{SM} \sum_{j=1}^{m} \ln p(s_{i,j}) \qquad (1)$$

is minimized for the best word $\hat{W}$, with i chosen for each j such that $s_{i,j}$ spells $\hat{W}$ and $p(\hat{W})$ being the language model probability for $\hat{W}$. w_{LM}, w_{SM} are respective weights for the language model and shape model. A more optimal model might use a different value of w_{LM} for each word length.

If we let $r = p(W)/p(W_{TC})$, and assume a single shape error in W_{TC}, then the language model can correct it by replacing a single top-choice shape $s_{TC,k}$ in position k in word W_{TC}, with a lower choice $s_{LM,k}$ in word W, provided that:

$$\ln r > \frac{w_{SM}}{w_{LM}} \left(\ln \frac{p(s_{TC,k})}{p(s_{LM,k})} + \sum_{j \ne k} \ln p(s_{ij}) \right) \quad . \qquad (2)$$

This model suggests that it would be interesting to explore the correcting power of the language model as a function of $\ln(r)$ i.e. the log of frequency ratio between correct and incorrect words. For example consider W= 'the,' then the language model can correct an error, such as 'thc' if ln p('the')/p('thc') is enough to overcome the classifier confidence of 'c' against 'e' in (2).

C. Binary n-gram Dictionary Language Model

With a binary dictionary, the language model can tell if a word is included in the dictionary (IID), or out of dictionary (OOD). No frequency information is available. To combine the language model with the shape classifier, an OCR system such as Tesseract[11] will search the classifier results to find the most classifier-probable IID word. A weight w_{IID} controls the balance between W_{TC} and the most probable IID word. With the simplified classifier model described above, the binary dictionary is completely disabled if $w_{IID} < p(s_{tc})/p(s_i)$ and wildcarding otherwise, since all non-top-choice shapes are equally probable. We therefore ignore w_{IID} and allow wildcarding. If the wildcards produce d IID words, one of which is correct, then the model chooses randomly between them and obtains the correct word with probability $1/d$.

This model suggests that the correcting power of the binary dictionary will be dependent on the size of the dictionary, so we explore the correcting power of the dictionary as a function of dictionary size.

IV. Experiments

Experiments were performed on the Google Books n-gram English corpus, which was split by alternate decades. Books published in odd decades were used to create D, and books published in even decades were used for C.

A. Word n-gram Frequency Models

For the frequency-based language models, the *correctability ratio* at position $j \in [1, m]$ of a word W is:

$$r_j = min(p(W)/p(\hat{W})) : \hat{W} \in \Omega(W, j), \hat{W} \neq W,$$
$$r_i = p(W)/\varepsilon \; if \; \Omega(W, j) = \{W\} \quad (3)$$

where $\Omega(W, j)$ is the set of words produced by a wildcard substitution over D at position j in word W, and ε is the probability of a word that occurs once in D. In words W is at least r_j times as frequent as its nearest wildcard competitor.

Analogously, the *corruptability ratio*,

$$r_c = max(p(\hat{W})/p(W)) : j \in [1, m], \hat{W} \in \Omega(W, j) \quad (4)$$

defines the margin by which an incorrect, but more probable word beats the correct word. For each corpus word W with frequency w, the final word error rate histogram for shape error rate accumulates the values shown in Table I as a function of *log r,* and shows the trade-off between correctability and corruptability.

This analysis ignores the possibility that the shape classifier makes an error at one position in W, and the language model wildcards at another position, making a more frequent word than correcting the original shape classifier error. This is a relatively low probability event, but makes the final error an under-estimate.

B. Binary Language Model

In the binary models, corrections are only attempted when W_{TC} is OOD. At position j of a word W of length m shapes, we estimate the probability of IID hallucination, where a shape classifier error creates an incorrect dictionary word, as $h=d/N_s$ where d=|Ω(*W, j*)|, being the fraction of the shape set that can produce an IID word. A word with a single shape error is *correctable* with probability $p_c=(1-h)/d$, provided that $W \in \Omega(W, j)$, and assuming that the language model chooses randomly among $\Omega(W, j)$.

If $\exists \; \hat{W} \in \Omega(W, j), \hat{W} \neq W : p(W) < p(\hat{W})$, then W will be corrupted when the dictionary contains $\hat{W}$ but not W, i.e. the size threshold lies between $p(W)$ *and* $p(\hat{W})$. We therefore require the max frequency of any incorrect wildcard word:

$$x_c = max(\log p(\hat{W})) : j \in [1, m], \hat{W} \in \Omega(W, j), \hat{W} \neq W \quad (5)$$

The x-axis of the histograms in Fig. 2 is the log of the size of the dictionary, but the word error rate histogram for shape error rate p_e accumulates values shown in Table II as a function of $x = log\ p(\hat{W}) : \hat{W} \in \Omega(W, j)$ as a histogram bucket proxy for dictionary size, and the actual dictionary size is calculated using the same buckets.

C. Problems of Coverage

The filtered corpus includes only n-grams that occur in at least 40 different books[6][13], and the coverage of D over C exceeds 99.99%, even on the 3-grams. This is unrealistic in real OCR data, so the experiments were re-run using the raw corpus. This reduced the coverage for the 1-, 2- and 3-grams to 99.76%, 97.55%, and 88.66% respectively.

TABLE I. WORD ERROR RATE HISTOGRAM FOR FREQUENCY-BASED MODELS

Value	Range	Description
$w(1-(1-p_e)^n)$	$0 \le r \le \infty$	Probability of one or more shape classifier errors.
$w(1-p_e)^n$	$0 \le r \le r_c$	Top-choice classifier correct, but corrupted by language model.
$-wp_e(1-p_e)^{n-1}$	$0 \le r \le r_j$	Single error at position j occurs and is corrected. Applied to each $j \in [1, n]$.

TABLE II. WORD ERROR RATE HISTOGRAM FOR BINARY MODELS

Value	Range	Description
$w(1-(1-p_e)^n)$	$-\infty \le x \le 0$	Probability of one or more shape classifier errors.
$w(1-p_e)^n$	$log\ p(W) < x < x_c$	Top-choice classifier correct, but corrupted by language model.
$-w(1-h)p_e(1-p_e)^{n-1}/d$	$-\infty \le x \le log\ p(W)$	Single error at position j occurs and is corrected. Applied to each $j \in [1, n]$.

D. Binary-Frequency Hybrid

A clear result from the frequency-based language model is that correct words are corrupted too frequently. As the error rate of the shape classifier falls, this corruption effect starts to dominate unless the weight of the language model is turned down so low that it hardly has any effect. (See Fig. 1.) Conversely the binary model benefits more from a larger dictionary as the error rate falls. This suggests a simple binary hybrid model, in which the language model does not attempt to change an IID W_{TC}, whether hallucination or correct, but when it does correct a word, uses the frequency to choose the result. The corruptability ratio is slightly different, since the word being corrected is always OOD:

$$r_c = max(p(\hat{W})/\varepsilon) : j \in [1, n], \hat{W} \in \Omega(W, j) \quad , \quad (6)$$

where ε is the frequency associated with an OOD word - corresponding to a count of 1 in the training corpus. As with the frequency-based model, the histogram accumulates values as a function of *log r*, shown in Table III.

V. RESULTS

Fig. 1 shows the final word error rates of the n-gram frequency models, for shape classifier error rates of 2%, 1% and 0.2%, as a function of *log r* from (2). Fig. 1(a), (b), (c) use a word 1-, 2- and 3-gram frequency model respectively, on the filtered corpus, and Fig. 3 shows the results for the raw corpus. At the y-axis, the language model substitutes a more frequent word, at the rate of the shape error, if one is available within a single wildcard substitution. As r increases, the corruption rate decreases, but also the correction rate, until at $r=\infty$, the output error rate is just the input shape classifier error rate. At low input error rate, the optimum point is little better than the input error rate, as the corruption rate falls off asymptotically, due to the long tail of the Zipf distribution, to a non-zero value: *Zipf's Wall*.

TABLE III. WORD ERROR RATE HISTOGRAM FOR BINARY HYBRID MODELS

Value	Range	Description
$w(1-(1-p_e)^n)$	$0 \le r \le \infty$	Probability of one or more shape classifier errors.
$w(1-p_e)^n$	$0 \le r \le r_c$	Top-choice classifier correct, but corrupted by language model.
$-w(1-h)p_e(1-p_e)^{n-1}$	$0 \le r \le r_j$	Single error at position j is OOD and is corrected. Applied to each $j \in [1, n]$.

Fig. 2 shows the final word error rates of the binary models, for shape classifier error rates of 2%, 1% and 0.2%, as a function of ln(dictionary size). Fig. 2(a), (b), (c) use a word 1-, 2- and 3-gram binary model respectively, and similarly Fig. 4 shows the results for the raw corpus. The corruption rate decreases with a larger dictionary, as more words are considered IID and not changed, but the correction rate also decreases, since more wildcard words can be found. For the 2- and 3-gram models, the optimum error rate is only obtained at the highest dictionary size. The sudden drop in Fig. 4(c) is caused by the addition of the words with a count of 1 in the Dictionary corpus *D*.

Fig. 5 shows the final word error rates of the hybrid model, for shape classifier error rates of 2%, 1% and 0.2%, as a function of log r. Fig. 5(a), (b), (c) use a word 1-, 2- and 3-gram model respectively. A summary of the minimum error points for all the models is given in Table IV. Table IV and Fig. 5 show the advantage of the simple non-linear rule that an IID word should not be touched.

VI. Conclusion: Speech vs OCR and FurtherWork

The fundamental limitation in the frequency models is the assumption that there is a close relationship between the probability that a word occurs in the language (given some current context for *n*-grams with $n>1$) and the probability that a word is correct. If the classifier (shape for OCR, or acoustic for speech) is weak, then the most frequent word is the best guess because it wins on average. A weak classifier will also frequently hallucinate incorrect dictionary words, so it is much more powerful to predict a word based on the previous *n*-1 words. When the classifier is fundamentally more accurate, the balance changes, and the most probable word is no longer the best guess, *even in the context of the previous n-1 words*. If the classifier top choice is an IID word, then it is much more likely to be correct than the most probable word within edit distance 1. This explains the dichotomy of approaches between speech and OCR.

Complex language models are highly beneficial at the high errors rates seen by acoustic classifiers for speech, but OCR shape classifiers are typically much more accurate (for Latin-based languages such as English), so language models are less effective. For languages where OCR is still at a higher error rate, such as Arabic and Hindi, powerful language models are still very useful.

Neither frequency-based language models nor the log-linear model can be declared dead for OCR however; the binary-frequency hybrid showed that the most frequent wildcard word is still an extremely good best-guess when W_{TC} is OOD, and it can still be expressed as a log-linear combination, using non-linear feature functions:

$$cost = -w_{LM} \ln p(\hat{W}) - w_{TC} f(\hat{W}) - w_{SM} \sum_i \ln p(s_{i,j}) \quad (7)$$

where $f(\hat{W}) = 0$ if $\hat{W}=W_{TC} \in D$, –1 otherwise.

A gross simplification of this experiment was the classifier error model, in which the language model has to guess from the entire character set. Going forward, noisy-channel models[14] specifically trained on the substitution and segmentation errors of a particular OCR shape classifier, such as e->c is more likely than o->x, would seem more appropriate than blind wildcarding. That should reduce the frequency of hallucinations of incorrect dictionary words, making the language model relatively more powerful.

The Google Books n-gram corpus is available for languages other than English, and it would be useful and interesting to compare results for other languages.

TABLE IV. Summary of Minima of Word Error Rates

Corpus		**Filtered**			**Raw**		
Shape error rate		2.00%	1.00%	0.20%	2.00%	1.00%	0.20%
Input word error rate		8.40%	4.20%	0.84%	9.10%	4.60%	0.91%
Model	n-gram	Minimal output word error rate (%)					
Freq.	1	4.54	2.50	0.65	5.01	2.76	0.73
Freq.	2	3.90	2.22	0.61	3.82	2.13	0.57
Freq.	3	2.71	1.65	0.52	3.23	1.72	0.43
Binary	1	4.73	2.58	0.60	5.77	3.17	0.77
Binary	2	3.69	1.80	0.35	5.08	2.56	0.51
Binary	3	1.78	0.82	0.15	3.83	1.89	0.37
Hybrid	1	2.87	1.37	0.26	4.64	2.28	0.45
Hybrid	2	1.28	0.56	0.10	2.00	0.94	0.18
Hybrid	3	0.65	0.24	0.03	1.82	0.86	0.16

Acknowledgment

The author would like to thank Rika Antonova, David Eger, Oded Fuhrmann, Dar-Shyang Lee and Ranjith Unnikrishnan for their valuable contributions to this paper.

References

[1] M. Bokser, "Omnidocument Technologies," Proc. IEEE vol. 80(7), Jul 1992, pp. 1066-1078, IEEE.

[2] L. R. Bahl, P. F. Brown, P. V. de Souza, R. L. Mercer, "A tree-based statistical language model for natural language speech recognition," IEEE trans. Acoustics, Speech and Signal Processing vol 37, Jul 1989, pp1001-1008.

[3] R. Kuhn, R. De Mori, "A cache-based natural language model for speech recognition," IEEE trans. Pattern Analysis and Machine Intelligence vol 12, Jun 1990, pp570-583.

[4] A. Lee, T. Kawahara, K. Shikano, "Julius – an open source real-time large vocabulary recognition engine," Proc. Eurospeech-2001, pp1691-1694.

[5] L. Zhidong, R. Schwartz, P Natarajan, I. Bazzi, J. Makhoul, "Advances in the BBN Byblos OCR system," Proc. 5th ICDAR, pp337-340, IEEE Sep 1999.

[6] Ngrams.googlelabs.com/info

[7] L. A. Adamic, "Zipf, power-laws, and pareto – a ranking tutorial," http://www.hpl.hp.com/research/idl/papers/ranking/ranking.html.

[8] S. Singh, The Code Book. Doubleday, 1989.

[9] T. K. Ho, G. Nagy, "OCR with no shape training," Proc. 15th ICPR, pp27-30, IEEE 2000.

[10] A. Kae, E. Learned-Miller, "Learning on the fly: font-free approaches to difficult OCR problems," Proc. 10th ICDAR, pp571-575, IEEE 2009.

[11] http://code.google.com/p/tesseract-ocr.

[12] F. J. Och, "Minimum error rate training in statistical machine translation," Proc. 41st Annual Meeting on Association for Computational Linguistics, pp160-167, ACL 2003.

[13] M. Jean-Baptiste, et al., "Quantitative analysis of culture using millions of digitized books," Science vol. 331, p176, 2011, doi: 10.1126/science.1199644.

[14] E. Brill, R. C. Moore, "An improved error model for noisy-channel spelling correction," Proc. 38th Annual Meeting on Association for Computational Linguistics, pp286-293, ACL, 2000.

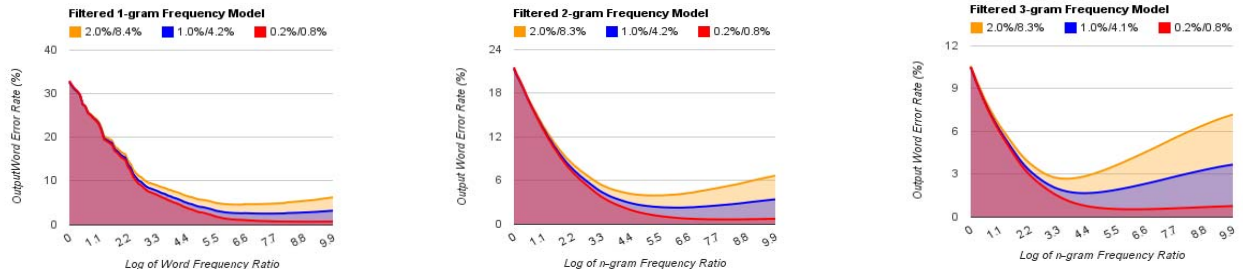

Figure 1. Output word error rate of (a) 1-gram, (b) 2-gram, (c) 3-gram Frequency models on the filtered corpus as a function of log of n-gram frequency ratio, with input shape classifier/word error rates of 2%/8.4%, 1%/4.2%, and 0.2%/0.8%.

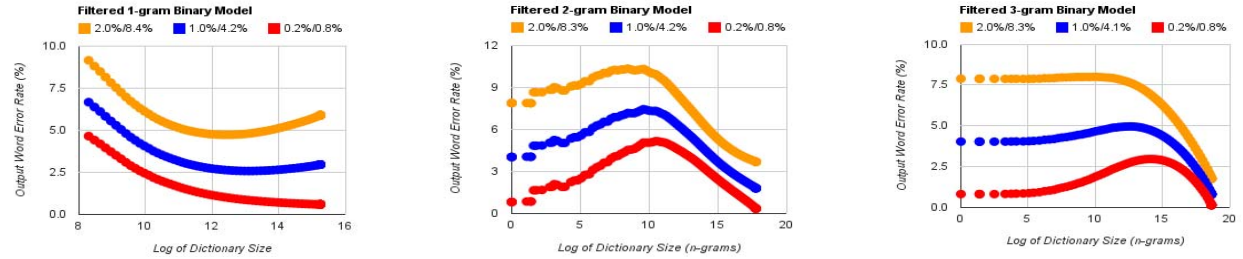

Figure 2. Output word error rate of (a) 1-gram, (b) 2-gram, (c) 3-gram Binary models on the filtered corpus as a function of log of n-gram dictionary size, with input shape classifier/word error rates of 2%/8.4%, 1%/4.2%, and 0.2%/0.8%.

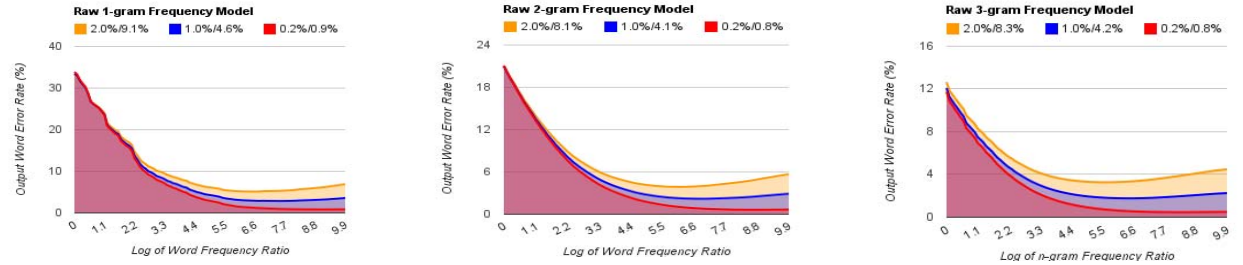

Figure 3. Output word error rate of (a) 1-gram, (b) 2-gram, (c) 3-gram Frequency models on the raw corpus as a function of log of n-gram frequency ratio, with input shape classifier/word error rates of 2%/9.1%, 1%/4.6%, and 0.2%/0.9%.

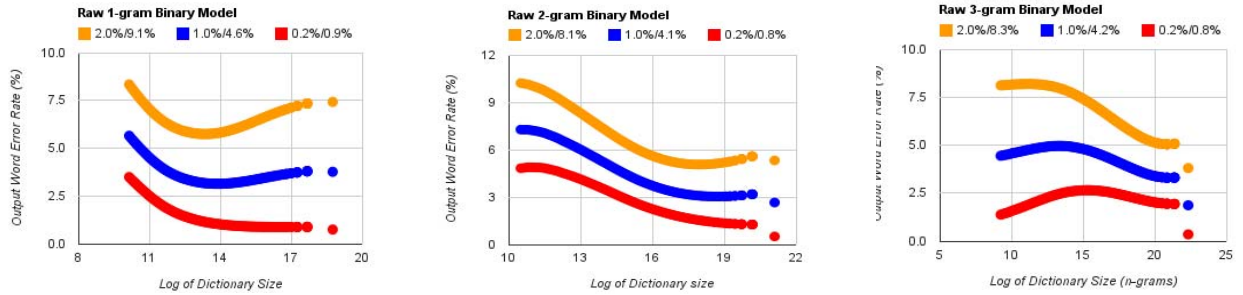

Figure 4. Output word error rate of (a) 1-gram, (b) 2-gram, (c) 3-gram Binary models on the raw corpus as a function of log of n-gram dictionary size, with input shape classifier/word error rates of 2%/9.1%, 1%/4.6%, and 0.2%/0.9%.

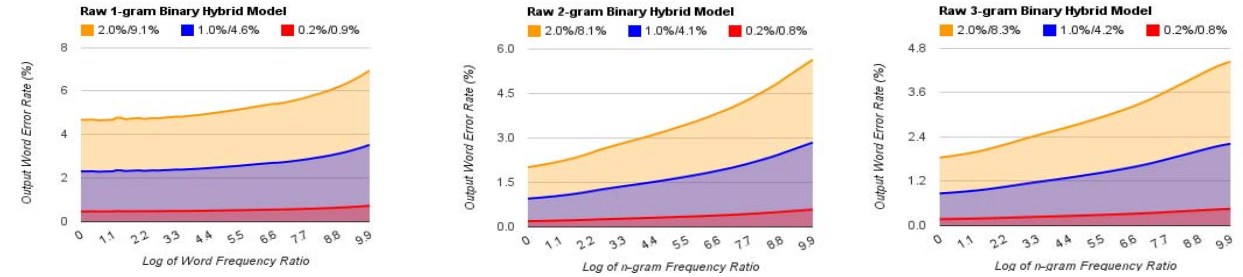

Figure 5. Output word error rate of (a) 1-gram, (b) 2-gram, (c) 3-gram Hybrid models on the raw corpus as a function of log of n-gram frequency ratio, with input shape classifier/word error rates of 2%/9.1%, 1%/4.6%, and 0.2%/0.9%.