What to Expect from Expected Kneser-Ney Smoothing

Michael Levit, Sarangarajan Parthasarathy, Shuangyu Chang

Microsoft, USA

{mlevit|sarangp|shchang}@microsoft.com

Abstract

Kneser-Ney smoothing on expected counts was proposed recently in [1]. In this paper we revisit this technique and suggest a number of optimizations and extensions. We then analyze its performance in several practical speech recognition scenarios that depend on fractional sample counts, such as training on uncertain data, language model adaptation and Word-Phrase-Entity models. We show that the proposed approach to smoothing outperforms known alternatives by a significant margin.

Index Terms: Language Modeling, Fractional Counts, Expected Kneser-Ney Smoothing

1. Introduction

Despite rapid advances of neural network language modeling, n-gram models remain at the center of most ASR systems, be it in the first-pass recognition or as part of a second-pass rescoring setup in combination with other model types. Kneser-Ney smoothing and its variants [2, 3] have been de facto standard smoothing for n-gram LMs for decades. However, when dealing with LM training on uncertain data, researchers had to switch to other techniques (such as fractional Witten-Bell [4]), since there was no straightforward extension of KN-smoothing to fractional counts. Recently, a clever modification of KN smoothing was proposed that accommodated fractional counts [1]. Instead of dealing with fractional counts directly, the authors viewed them as the probability of observing an n-gram in a sample corpus and modified the KN smoothing algorithm to work with count expectations over training corpora.

There are several language modeling scenarios that rely on fractional counts of input samples and can therefore welcome this extension. For instance, unsupervised language model training and adaptation on ASR transcripts was shown to be beneficial as the amounts of unsupervised data increased [5]. The scores of the transcription hypotheses can also be used to improve LM quality via thresholding [6], sampling [7] or direct modification of the smoothing algorithm [8]. We therefore expect the algorithm from [1] to be helpful for this scenario. A similar argument can be applied to LM training on competing human transcriptions hypotheses. Another relevant scenario is adaptation where training material is projected on a number of pre-defined or latent topics, and for each of these topics a dedicated language model is built. Finally, fractional counting techniques apply naturally to Word-Phrase-Entity LMs [9], where the probability mass of training sentences is split among its possible representations in terms of phrases and entities. In this paper we analyze benefits of the expected KN on all these scenarios. We also discuss several modifications of the technique such as n-gram cutoffs, training on sentences with accumulated counts and n-best, and review expected version of the Witten-Bell smoothing.

2. Expected Kneser-Ney Smoothing

In this section we introduce KN smoothing on expected counts, closely following the material from [1]. In addition, in [10] we have published an in-depth derivation of the formulae below. First, recall the standard KN smoothing as a version of absolute discounting turning n-gram counts into:

$$\tilde{c}(\boldsymbol{u}w) = \begin{cases} c(\boldsymbol{u}w) - D, & \text{if } c(\boldsymbol{u}w) > 0 \\ \gamma'(\boldsymbol{u}) p^{\text{kn}}(w|\,\boldsymbol{u'}) & \text{otherwise} \end{cases} \quad (1)$$

that falls back on special lower order probability of the form:

$$p^{\text{kn}}(w|\,\boldsymbol{u'}) = \frac{n_{1+}(\bullet\,\boldsymbol{u'}w)}{n_{1+}(\bullet\,\boldsymbol{u'}\,\bullet)}. \quad (2)$$

In the above, D is estimated via Leave-One-Out as

$$D = n_1/(n_1 + 2n_2)$$

with n_r standing for the number of n-grams in the corpus that occurred exactly r times (n_{r+} meaning "at least r") and $\bullet$ being "any word". Normalization factor $\gamma'(\boldsymbol{u})$ for context $\boldsymbol{u}$ is computed to assure stochastic conditions for probabilities. In addition, the Modified KN flavor [3] calls for separate estimates $D_r^{(n)}$ for different r (up to 3) and n-gram orders, recursive smoothing applied to KN-estimates and (optionally) replacing back-offs with interpolation. This smoothing technique has been repeatedly shown to outperform most of its competitors.

The basic idea behind expected KN (eKN) is to replace the counts with their expectations over the training corpus in which each n-gram instance would appear with the probability equal to the specified fractional count:

$$\tilde{p}(w|\,\boldsymbol{u}) = \frac{E[\tilde{c}(\boldsymbol{u}w)]}{E[c(\boldsymbol{u}\,\bullet)]} = \frac{E[\tilde{c}(\boldsymbol{u}w)]}{\sum_v E[c(\boldsymbol{u}w)]}. \quad (3)$$

This turns back-off probability in Eq. (2) into

$$p^{\text{kn}}(w|\,\boldsymbol{u'}) := \frac{E[n_{1+}(\bullet\,\boldsymbol{u'}w)]}{E[n_{1+}(\bullet\,\boldsymbol{u'}\,\bullet)]}. \quad (4)$$

Following definition of expectation, Eq. (1) becomes:

$$\begin{aligned} & E[\tilde{c}(\boldsymbol{u}w)] \\ = & \sum_{r>0} p(c(\boldsymbol{u}w)=r)(r-D) + p(c(\boldsymbol{u}w)=0)\gamma'(\boldsymbol{u})p^{\text{kn}}(w|\,\boldsymbol{u'}) \\ = & E[c(\boldsymbol{u}w)] - DP + p(c(\boldsymbol{u}w)=0)\gamma'(\boldsymbol{u})p^{\text{kn}}(w|\,\boldsymbol{u'}) \end{aligned} \quad (5)$$

where the last step introduced average discount

$$DP := p(c(\boldsymbol{u}w)>0) * D \quad (6)$$

which, for the modified KN, turns into

$$DP := p(c(\boldsymbol{u}w)=1)*D_1 + p(c(\boldsymbol{u}w)=2)*D_2 + p(c(\boldsymbol{u}w)\geq 3)*D_3 \quad (7)$$

(for brevity, we will omit the n-gram index). Going over to probabilities, and introducing a new normalization factor $\gamma(\boldsymbol{u})$, Eq. (5) becomes:

$$\tilde{p}(w|\,\boldsymbol{u}) = \frac{E[c(\boldsymbol{u}w)] - DP}{\sum_v E[c(\boldsymbol{u}v)]} + p(c(\boldsymbol{u}w) = 0)\gamma(\boldsymbol{u})p^{\text{kn}}(w|\,\boldsymbol{u'}) \tag{8}$$

To compute statistics needed in Eq. (5), [1] notice that if an n-gram occurs K times in a corpus with respective fractional counts p_k, by taking this counts as probabilities, the expectation of the total count of this n-gram is:

$$E[c(\boldsymbol{u}w)] = \sum_{k=1}^{K} p_k. \tag{9}$$

Probabilities of specific cumulative frequencies r given k fractional instances can be computed recursively with:

$$p(c(\boldsymbol{u}w){=}r) = s_K^r \tag{10}$$

where

$$s_k^r = \begin{cases} s_{k-1}^r(1-p_k) + s_{k-1}^{r-1}p_k, & \text{if } 0 \le r \le k \\ 1, & \text{if } k = r = 0 \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

The complexity remains linear in k since $r \le 4$. The expected value of the number of all n-grams with cumulative frequency r is computed by summing over all n-grams:

$$E[n_r] = \sum_{\boldsymbol{u}w} p(c(\boldsymbol{u}w){=}r) \tag{12}$$

which, in turn, allows us to define the expectation-version of the absolute discounts. Here, for the modified version:

$$\begin{array}{lcl} D & = & E[n_1]/(E[n_1] + 2E[n_2]) \\ D_1 & = & 1 - 2D\frac{E[n_2]}{E[n_1]} \\ D_2 & = & 2 - 3D\frac{E[n_3]}{E[n_2]} \\ D_3 & = & 3 - 4D\frac{E[n_4]}{E[n_3]} \end{array} \tag{13}$$

For a recursive version of the eKN smoothing, observe that, just like Eq. (3), probability in Eq. (4) can be viewed as a fraction of two expected counts: "KN-counts". This and a simple to prove equivalence of probabilities of zero occurrences for original and KN-counts, allows us to write:

$$\tilde{p}(w|\,\boldsymbol{u}) := \frac{E[\hat{c}(\boldsymbol{u}w)] - DP}{\sum_v E[\hat{c}(\boldsymbol{u}v)]} + p(\hat{c}(\boldsymbol{u}w){=}0)\gamma(\boldsymbol{u})\tilde{p}(w|\,\boldsymbol{u'}) \tag{14}$$

with

$$E[\hat{c}(\boldsymbol{u}w)] = \begin{cases} E[c(\boldsymbol{u}w)] = \sum_{k=1}^{K} p_k & \text{if } |\,\boldsymbol{u}w\,| = N \\ & \text{or } \boldsymbol{u}w = \text{"<s>}\ldots\text{"} \\ E[c^{\text{kn}}(\boldsymbol{u}w)] = \sum_v p(c(v\,\boldsymbol{u}w){>}0) & \text{otherwise} \end{cases}$$

For more details, please see [1, 10].

3. Proposed Solutions and Extensions

Next, we describe a number of extensions and practical solutions for the eKN framework making it usable for realistic scenarios. To compute normalization factor $\gamma(\boldsymbol{u})$, we observe that sum of Eqs. (14) over w should be 1 for all $\boldsymbol{u}$, resulting in:

$$\gamma(\boldsymbol{u}) = \frac{1 - \sum_w \frac{E[\hat{c}(\boldsymbol{u}w)] - DP}{\sum_v E[\hat{c}(\boldsymbol{u}v)]}}{\sum_w p(\hat{c}(\boldsymbol{u}w){=}0) * \tilde{p}(w|\,\boldsymbol{u'})} \tag{15}$$

First, consider the case, where no n-gram cutoffs are applied. In this case, Eq. (15) can be simplified as:

$$\begin{array}{rcl} \gamma(\boldsymbol{u}) & = & \frac{\sum_w DP}{\sum_w E[\hat{c}(\boldsymbol{u}w)] * \sum_w p(\hat{c}(\boldsymbol{u}w){=}0)\tilde{p}(w|\,\boldsymbol{u'})} \\ & = & \frac{\sum_{w:c(\boldsymbol{u}w)>0} DP}{\sum_{w:c(\boldsymbol{u}w)>0} E[\hat{c}(\boldsymbol{u}w)]} * \\ & & \frac{1}{1 - \sum_{w:c(\boldsymbol{u}w)>0}(1 - p(\hat{c}(\boldsymbol{u}w){=}0))\tilde{p}(w|\,\boldsymbol{u'})} \end{array} \tag{16}$$

where the second transition is due to decomposition:

$$\begin{array}{rcl} \sum_w p(\hat{c}(\boldsymbol{u}w){=}0)\tilde{p}(w|\,\boldsymbol{u'}) & = & \\ \sum_{w:c(\boldsymbol{u}w)>0} p(\hat{c}(\boldsymbol{u}w){=}0)\tilde{p}(w|\,\boldsymbol{u'}) + 1 - \sum_{w:c(\boldsymbol{u}w)>0} \tilde{p}(w|\,\boldsymbol{u'}) & = & \\ 1 + \sum_{w:c(\boldsymbol{u}w)>0} \tilde{p}(w|\,\boldsymbol{u'})\,(\hat{p}(c(\boldsymbol{u}w){=}0) - 1) & & \end{array} \tag{17}$$

This allows for easy summation over just the n-grams $\boldsymbol{u}w$ that were observed in training.

3.1. Cutoffs and Pruning

Practical applications often impose restrictions on LM size. One of the ways to reduce the size is through introducing cutoff thresholds on n-gram counts. In our experiments, we confirmed that just like for standard KN, eKN provides best results when these cutoffs are applied to KN-counts. Like the counts themselves, these thresholds can be real-valued. With some n-grams eliminated, the numerator term of Eq. (15) needs to be revisited. This time, cumulative counts

$$\hat{C}(\boldsymbol{u}) := \sum_w \hat{c}(\boldsymbol{u}w) \tag{18}$$

need to be maintained separately (at least for the KN-counts) and the formula for γ becomes:

$$\gamma(\boldsymbol{u}) = \frac{E[\hat{C}(\boldsymbol{u})] - \sum_w (E[\hat{c}(\boldsymbol{u}w)] - DP^{(n)})}{E[\hat{C}(\boldsymbol{u})] * \sum_w p(\hat{c}(\boldsymbol{u}w){=}0)\tilde{p}(w|\,\boldsymbol{u'})} \tag{19}$$

A similar adjustment needs to be made to other formulae as well; for instance, in the denominator of Eq. (14).

The other technique to reduce the model size is entropy-based pruning which is independent of the smoothing method and can be applied to the resulting LM without restrictions. Nonetheless, mindful of possible losses due to aggressive entropy-based pruning of KN-trained LMs [11], for our experiments, we kept the pruning threshold to be less than 1e-8 and observed little to no degradation due to pruning while still drastically reducing model size.

3.2. Compact Representation

Realistic corpora often contain multiple instances of the same examples. Their counts can be easily incorporated in the computation of all standard smoothing techniques, including KN, but for eKN a variation of Eq. (11) needs to be applied to produce cumulative statistics s_K^r. If m instances of an n-gram occurred with probability p, the probability of observing this n-

gram $r = 0, 1, 2, \ldots$ times is:

$$\begin{aligned} s_m^0 &= (1-p)^m \\ s_m^1 &= m(1-p)^{m-1} * p \\ s_m^2 &= \frac{m(m-1)}{2}(1-p)^{m-2} * p^2 \\ &\ldots \end{aligned} \tag{20}$$

which can be efficiently computed via recursion. If we have two instances, one occurring m_1 times with probability p_1 and the other occurring m_2 times with probability p_2, corresponding statistics $s_{m_i}^r$ can be combined in an obvious way:

$$\begin{aligned} s_{m_1+m_2}^0 &= s_{m_1}^0 * s_{m_2}^0 \\ s_{m_1+m_2}^1 &= s_{m_1}^1 * s_{m_2}^0 + s_{m_1}^0 * s_{m_2}^1 \\ s_{m_1+m_2}^2 &= s_{m_1}^2 * s_{m_2}^0 + s_{m_1}^1 * s_{m_2}^1 + s_{m_1}^0 * s_{m_2}^2 \\ &\ldots \end{aligned} \tag{21}$$

By induction, this can be applied to all occurrences of the n-gram in the corpus eventually resulting in s_K^r.

3.3. N-best

Training on sets of alternative sentence representation is a common strategy in language modeling. For instance, it is known that unsupervised training on ASR n-best hypotheses can boost language model quality for an existing application [7]. Unlike independent training sentences, n-grams from n-best alternatives of the same sentence can't be considered independently. For instance, if there are two alternatives with (posterior) weights:

hello world 0.8
hello dolly 0.2

they do not contribute to the unigram *"hello"* twice (once with p=0.8 and another time with p=0.2), but rather a single time with p=1.0. The difference is significant: in the first case, there is a non-zero probability that a sample corpus drawn from the training data will have no *"hello"*' at all. In the second case, the *"hello"* is guaranteed to appear in the sample corpus . The proper way to handle such input is to operate on lattices [8]. In our experiments, we use a simpler representation that turns n-best strings into an n-gram trie.

3.4. Interpolated Version (with Cutoffs)

In [3] it was noted that interpolated version of the KN smoothing generally leads to better results. The difference to the original KN is in spreading saved probability mass across all words in the context $\boldsymbol{u}$ and not just the words that never occurred in it. This version also happens to be easier to implement [1]: we just need to drop $p(c(\boldsymbol{u}w) = 0)$ from all formulae for $\gamma(\boldsymbol{u})$ and $\tilde{p}(w|\,\boldsymbol{u})$. For instance, instead of Eq. (14) there will be:

$$\tilde{p}(w|\,\boldsymbol{u}) := \frac{E[\hat{c}(\boldsymbol{u}w)] - DP}{E[\hat{C}(\boldsymbol{u})]} + \gamma(\boldsymbol{u})\tilde{p}(w|\,\boldsymbol{u'}) \tag{22}$$

and in place of Eq. (19):

$$\begin{aligned} \gamma(\boldsymbol{u}) &= \frac{E[\hat{C}(\boldsymbol{u})] - \sum_w (E[\hat{c}(\boldsymbol{u}w)] - DP)}{E[\hat{C}(\boldsymbol{u})] * \sum_w 1 * \tilde{p}(w|\,\boldsymbol{u'})} \\ &= \frac{E[\hat{C}(\boldsymbol{u})] - \sum_w (E[\hat{c}(\boldsymbol{u}w)] - DP)}{E[\hat{C}(\boldsymbol{u})]} \end{aligned} \tag{23}$$

Like the integer-valued KN, the interpolated version of the expected KN can be also expressed as a back-off language model (ARPA format) with discounted logscores for n-gram $\boldsymbol{u}w$

$$\alpha(\boldsymbol{u}w) = \log(\tilde{p}(w|\,\boldsymbol{u}) + \gamma(\boldsymbol{u}) * \tilde{p}(w|\,\boldsymbol{u'})) \tag{24}$$

and the back-off penalty associated with context $\boldsymbol{u}$

$$\beta(\boldsymbol{u}) = \log(\gamma(\boldsymbol{u})) \tag{25}$$

3.5. Expected Witten-Bell

Similar to eKN, we can define expectation-based conditional probability for Witten-Bell [12] smoothing (eWB):

$$\begin{aligned} \tilde{p}(w|\,\boldsymbol{u}) &= \sum_{r>0} p(c(\boldsymbol{u}w) = r) \frac{c(\boldsymbol{u}w)}{\sum_v c(\boldsymbol{u}v) + n_{1+}(\boldsymbol{u}\bullet)} \\ &+ p(c(\boldsymbol{u}w) = 0)\gamma(\boldsymbol{u})\tilde{p}(w|\,\boldsymbol{u'}) \end{aligned} \tag{26}$$

As in Eq. (5), we again recognize an expectation term $c(\boldsymbol{u})$ in the first addend. However, taking expectation inside a fraction is only a first order approximation in this case:

$$\begin{aligned} \tilde{p}(w|\,\boldsymbol{u}) &\approx \frac{E[c(\boldsymbol{u}w)]}{\sum_v E[c(\boldsymbol{u}v)] + E[n_{1+}(\boldsymbol{u}\bullet)} \\ &+ p(c(\boldsymbol{u}w) = 0)\gamma(\boldsymbol{u})\tilde{p}(w|\,\boldsymbol{u'}). \end{aligned} \tag{27}$$

As before, $\gamma(\boldsymbol{u})$ can be obtained via normalization [10].

4. Applications and Experiments

To evaluate the benefits of eKN and its interpolated version ieKN, we focused on three scenarios that depend on fractional counts. We compare perplexity numbers of eKN against several following baselines: fractional Witten-Bell (fWB), expected WB (eWB) and interpolated stochastic KN (isKN). As an additional baseline, we experimented with a number of methods implemented in OpenGRM [13] including fractional counting as described in [8], and report the best perplexities obtained.

4.1. Uncertain Data

Our first corpus is a collection of alternative transcriptions as well as n-best recognition results for 115K utterances from the Cortana domain. Each utterance has up to 5 alternative transcriptions obtained using the crowdsourcing approach (H) from [14] and up to 5 unsupervised ASR transcriptions (A). Each transcription comes with a confidence score which we normalize to sum up to 1.0 over all alternatives (including rejection) for each modality. For LM training, we have a choice between using only the highest scoring alternative with and without weights (e.g. H1 and H1-W for human transcriptions) and considering the entire set of weighted 5-bests (e.g. A5-W for ASR). For each corpora, we train 4-gram LMs using each of the smoothing techniques described earlier. The test corpus is a collection of highest-scoring transcriptions for 8.5K utterances from the same domain. Results are shown in Table 1.

First, note that fractional and expected techniques are identical for weightless 1-best training where they fall back to their integer-valued variants. Second, observe that perplexities of LMs trained on 5-best strings are higher than ones trained on 1-best hypotheses. The primary reason for this anomalous result is that these numbers do not account for OOVs in perplexity computation and 5-best LMs have much lower OOV rates than 1-best LMs. To make comparison between setups with different vocabularies fair, a very conservative estimate $p(<\text{unk}>)$=1e-6 (same as the lowest probability actually in the LMs) can be

Table 1: *Effect of smoothing techniques for 1-best and n-best training material.*

	fWB	eWB	OpenGRM	eKN	ieKN
H1	73.2	73.2	61.6	69.8	61.4
H1-w	73.2	73.4	65.8	69.7	61.4
H5-w	73.2	74.1	66.2	69.4	62.0
A1	72.8	72.8	61.8	69.8	61.7
A1-w	72.1	73.1	65.5	67.0	61.3
A5-w	78.2	75.8	71.5	67.4	63.4

introduced. With that, the benefits of n-best training become evident. For instance, for the ASR setup with ieKN we then get: ppx(A1)=76.6 and ppx(A5-w)=74.1. In the case of manual transcriptions, the effect is much weaker because most of the time, only a single transcription alternative is available per utterance. Most importantly, we observe that eKN, and in particular its interpolated variant ieKN significantly outperform other smoothing methods, including the strong OpenGRM baseline. The expected version of the WB smoothing leads to perplexity numbers comparable with the fractional WB.

4.2. Split-and-Merge Technique for LM Adaptation

Our next experiment is focused on language model adaptation which was also the subject of the eKN experiments in [1]. There, the training material was soft-partitioned into in-domain and out-of-domain subsets according to a pair of in- and out-of-domain seed LMs. eKN was used to train a language model on the in-domain partition after each sentence was assigned a fractional score reflecting how much higher the LM-log-scores of the in-domain seed LM for this sentence were compared to out-of-domain.

In this paper, we extended this approach in several ways. First, we trained 5 seed LMs that included message dictation, Command-and-Control utterances, Wikipedia, Xbox-related searches as well as generic voice search (VS). Next, we took a collection of five unseen training corpora (news, text web search, Twitter, Cortana recognition results and Cortana manual transcriptions) with a total of about 50M sentences, and computed their LM scores in all 5 seed LMs. This allowed us to project each input sentence on each of the 5 LMs using relative scores as projection values. These relative scores played the role of fractional counts. They were either fed directly to the fractional/expected smoothing techniques or dealt with via stochastic sampling before being made subject to standard smoothing algorithms. To speed up the training, counts were associated with unique sentences (cf. Section 3.2). Next, the 25 trained LMs were linearly interpolated using context-depended interpolation weights [15] optimized for one of 5 target domains: Command-and-Control, Xbox, message dictation, VS and TED talks. In other words, with our split-and-merge strategy, we decomposed all training sets into subsets and re-assembled those to best fit target domains. All language models (before and after interpolation) were pruned with threshold 1e-10 to contain on the order of 100M n-grams and share a common dictionary of 300K words. The models were then evaluated on tests from the 5 target domains. Results are shown in Table 2. An additional baseline iKN-all shows perplexities that were obtained by directly merging all input corpora (using EM-optimized weights) without prior decomposition.

In addition to seeing the obvious benefit of decomposition, we observe that eKN smoothing consistently outperforms other smoothing techniques, although its advantage against stochastic KN is only moderate. This is in contrast to a single-step adaptation from [1] lacking the merging stage. Indeed, when comparing perplexities of LMs trained on individual projections of input corpora, ieKN outperformed isKN by about 12% on average with conservative estimate $p(<\text{unk}>)$=1e-8. We therefore conclude that benefits of better smoothing techniques get diluted when the resulting models are merged with other LMs. Also, notice that in this setup, expected WB achieved consistent improvements over fractional WB. A similar experiment was carried out with other training corpora and seed LMs, including LDA-based seeds like in [16] with analogous results.

Table 2: *Different smoothing techniques for split-and-merge adaptation strategy.*

	iKN-all	fWB	eWB	isKN	ieKN
C&C	66.1	40.4	39.6	38.0	37.3
Xbox	93.3	66.7	64.5	62.1	61.2
dictation	114.2	90.4	86.9	82.4	80.1
VS	142.2	138.3	137.4	129.7	128.3
TED	341.3	241.5	236.4	209.5	204.0

Table 3: *Different smoothing techniques for split-and-merge adaptation strategy.*

	fWB	ieKN
n-grams WPE	37.6	34.8
n-grams WPE + RNN WPE	32.4	32.2

4.3. Word-Phrase-Entity LMs

Our last LM scenario that depends on fractional input weights are WPE-LMs from [9]. In WPE LMs for each sentence its alternative parses in terms of stable word phrases and named entities are generated. The parses are scored according to their LM scores in the previous iteration LM and contribute to the next iteration LM according to these scores. By nature, the relative contributions are fractional and in the past, we used to employ fractional WB. Now, we can replace it with expected KN smoothing. The resulting perplexity numbers for WPE n-gram LMs as well as these LMs interpolated with WPE RNN [17, 18], are shown in Table 3. Note that unlike [18] where the point of comparison was the advantage of the WPE technique, these perplexity numbers are provided w/o taking unknown words into consideration because vocabularies are identical. Again, we see 7.5% improvement due to superior smoothing method, but this advantage mostly disappears after interpolation.

5. Conclusion

We presented several extensions and optimizations for the Kneser-Ney smoothing on expected counts first introduced in [1]. They included training on compact corpus representations, weighted alternative representations of sentences, support for cutoff thresholds and training Witten-Bell on expected counts. We then investigated how this smoothing technique compares to traditional alternatives in three practical scenarios: training on uncertain data, split-and-merge technique for language model adaptation and training Word-Phrase-Entity models. Our conclusions are that KN on expected counts consistently outperforms other smoothing techniques on fractional counts, although the advantage gets diluted when the produced models are interpolated with other LMs (n-grams or NNLMs).

6. References

[1] H. Zhang and D. Chiang, "Kneser-Ney Smoothing on Expected Counts," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2014, pp. 765–774.

[2] R. Kneser and H. Ney, "Improved Backing-off for M-gram Language Modeling," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1. IEEE, 1995, pp. 181–184.

[3] S. F. Chen and J. Goodman, "An Empirical Study of Smoothing Techniques for Language Modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–394, 1999.

[4] A. Stolcke, "SRILM – an Extensible Language Modeling Toolkit," in *7th International Conference on Spoken Language Processing*, 2002.

[5] M. Bacchiani and B. Roark, "Unsupervised Language Model Adaptation," in *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, vol. 1. IEEE, 2003, pp. I–I.

[6] R. Gretter and G. Riccardi, "On-line Learning of Language Models with Word Error Probability Distributions," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, vol. 1. IEEE, 2001, pp. 557–560.

[7] M. Bacchiani, M. Riley, B. Roark, and R. Sproat, "MAP adaptation of Stochastic Grammars," *Computer Speech & Language*, vol. 20, no. 1, pp. 41–68, 2006.

[8] V. Kuznetsov, H. Liao, M. Mohri, M. Riley, and B. Roark, "Learning N-gram Language Models from Uncertain Data," in *Interspeech*, 2016.

[9] M. Levit, S. Parthasarathy, S. Chang, A. Stolcke, and B. Dumoulin, "Word-Phrase-Entity Language Models: Getting More Mileage out of N-grams," in *Proc. Interspeech*. Singapore: ISCA - International Speech Communication Association, September 2014.

[10] M. Levit, P. Parthasarathy, and S. Chang, "How to Use Expected Kneser-Ney Smoothing on Text Input with Fractional Counts," Microsoft Corporation, Technical Report MSR-TR-2018-16, 2018. [Online]. Available: https://www.microsoft.com/en-us/research/publication/n-gram-smoothing-expected-fractional-counts

[11] C. Chelba, T. Brants, W. Neveitt, and P. Xu, "Study on Interaction between Entropy Pruning and Kneser-Ney Smoothing," in *11th Annual Conference of the International Speech Communication Association*, 2010.

[12] I. H. Witten and T. C. Bell, "The Zero-Frequency Problem: Estimating the Probabilities of Novel Events in Adaptive Text Compression," *IEEE transactions on information theory*, vol. 37, no. 4, pp. 1085–1094, 1991.

[13] B. Roark, R. Sproat, C. Allauzen, M. Riley, J. Sorensen, and T. Tai, "The OpenGrm Open-Source Finite-State Grammar Software Libraries," in *Proceedings of the ACL 2012 System Demonstrations*. Association for Computational Linguistics, 2012, pp. 61–66.

[14] M. Levit, Y. Huang, S. Chang, and Y. Gong, "Dont Count on ASR to Transcribe for You: Breaking Bias with Two Crowds," in *Proceedings of Interspeech*, 2017.

[15] X. Liu, M. J. F. Gales, and P. C. Woodland, "Context Dependent Language Model Adaptation," in *Proc. Interspeech*, Brisbane, Australia, 2008.

[16] M. A. Haidar and D. O'Shaughnessy, "Topic N-gram Count Language Model Adaptation for Speech Recognition," in *Spoken Language Technology Workshop (SLT), 2012 IEEE*. IEEE, 2012, pp. 165–169.

[17] M. Levit, A. Stolcke, S. Chang, and S. Parthasarathy, "Token-Level Interpolation for Class-Based Language Models," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5426–5430.

[18] M. Levit, S. Parthasarathy, and S. Chang, "Word-Phrase-Entity Recurrent Neural Networks for Language Modeling," in *INTERSPEECH*, 2016, pp. 3514–3518.