UNSUPERVISED WORD INDUCTION USING MDL CRITERION

YU Hua

Interactive Systems Lab, Carnegie Mellon University, Pittsburgh
Email: `hyu@cs.cmu.edu`

ABSTRACT

Unsupervised learning of units (phonemes, words, phrases, etc.) is important to the design of statistical speech and NLP systems. This paper presents a general source-coding framework for inducing words from natural language text without word boundaries. An efficient search algorithm is developed to optimize the minimum description length (MDL) induction criterion. Despite some seemingly over-simplified modeling assumption, we achieved good results on several word induction problems.

1. INTRODUCTION

Imagine you are reading text like this: $\cdots$OKAYUHCOULDYOUTEL LMEWHATYOUTHINKCONTRIBUTESMOSTTOUHAIRPOLLUTI ON$\cdots$, and you know nothing about the language. How would you segment such a continuous stream of letters into words? Suppose you have no lexicon.

This is the task that word induction is trying to solve: automatically discovering words from *only* text with no word boundary information.

For languages like English, this is not a real problem, since words are nicely separated by blank spaces. However, for agglutinative languages such as Chinese and Japanese, there are no spaces between words, thus word induction becomes a real problem. For spoken English, unlike the printed form, there is usually no reliable acoustic evidence of word boundaries. Yet language acquisition researchers have long been wondering how children can segment speech and identify individual words from a state in which they do not know any words.

To a larger extent, we found the problem of word induction has much in common with other unit induction problems in natural language processing:

- acoustical unit induction: phonemes are widely used as the natural units for speech recognition. But researchers keep asking the question: what is the best phoneme set? The linguistically defined phoneme set might not be consistent with other components of the acoustic model, where statistical data-driven approach dominates. Recently people have begun working on acoustic sub-word units (ASWU) to try to automatically derive units from a continuous stream of speech data [6, 2].
- phrasal units induction: most language models are word based, although words are chosen as the modeling unit more for convenience than for any other reason. People have tried various ways to find phrases to build a language model. Phrase induction, again, is concerned with identifying recurrent patterns from a word stream.
- lexicon optimization: in speech recognition for languages such as Chinese and Japanese, it is not clear what should be included in the lexicon. If syllables are to be used, acoustic confusability increases because syllables are such short units. Language model also suffers when predictions have to be made from the previous N syllables. But if words or phrases are to be used, the lexicon will explode beyond the capacity of any real recognition system. Simply cutting down the lexicon size will cause high OOV (out of vocabulary) rates which also hurts performance. Here again we are facing the problem of choosing the right unit [9].
- syllabification: what constitutes a syllable? There is no clear definition, although syllables seem to be a very natural concept for human beings. We conjecture it is the recurrent nature that largely defines syllable, and unsupervised syllabification can be done in much the same way as word induction from text.

The choice of units is important to the design of statistical speech and NLP systems. Unsupervised learning of such units is consistent with the overall data-driven framework. In this paper, we propose a general source-coding framework (Section 2) for all unit induction problems, using MDL as the induction criterion. We have observed strong empirical evidence supporting this criterion. In Section 3, an algorithm is developed that directly optimizes the objective function. The model is applied to several word induction problems, including text extracted from the SWB(Switchboard) corpus and phonemic transcription of child directed speech. Despite some seemingly over-simplified assumptions, we showed word induction is feasible. Finally related works are reviewed.

2. SOURCE CODING MODEL & MDL CRITERION

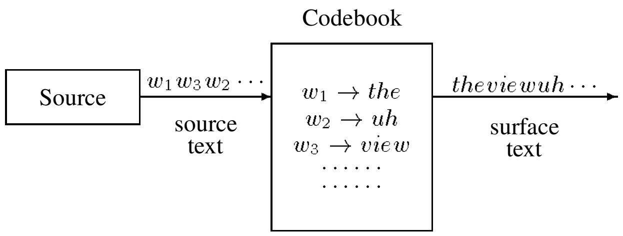

Figure 1: Source Coding Model

As Figure 1 illustrates, the text we observe in the surface form is generated from a “deep-form” text using a certain codebook. Neither the “deep-form” text nor the codebook (containing the definition of units) is directly observable. Our goal is to induce the codebook from the surface text only, i.e. a decipher problem.

In the case of written English without spaces, we observe only a character stream. The task is to segment it in a meaningful way so that we recover the original message $(w_1 w_3 w_2 \cdots)$ that the source is trying to communicate. Note a segmentation automatically defines a codebook: each unique character sequence in the segmentation is an entry in the codebook.

Two types of knowledge might help in the deciphering process: source properties and codebook properties. Codebook properties, for example, might include well-formedness of units, unit length distribution, etc.[1] But in general this is not the case. Source properties (our knowledge about the source) don't seem to be obviously computable at first glance. But if we acknowledge the source is not emitting pure gibberish, the source text should have a low entropy.

More formally, the optimal codebook should satisfy:

$$\begin{aligned} CB^* &= arg \max_{CB} L(O|CB)L(CB) \\ &= arg \min_{CB} -\log L(O|CB) - \log L(CB) \end{aligned}$$

where O is the observed stream of characters, $L(O|CB)$ measures the likelihood of the data, $L(CB)$ is the prior distribution on all codebooks. Or from the information theory point of view, $-\log L(O|CB)$ is the entropy of the source text, $-\log L(CB)$ is the entropy of the codebook. That is, we are seeking a codebook that offers an optimal tradeoff between the description length of O and its own complexity, measured in number of bits. This is exactly the MDL principle. By taking the codebook cost into account, it reduces the risk of over-fitting and provides better generalization capability than the ML (Maximum Likelihood) criterion.

Thinking in terms of compressing the message O, this basically says we need to transmit not only the encoded text, but also the codebook. We can either adopt a small codebook and transmit the message literally with little coding; or use a huge codebook that encodes the entire message as a single entry, thus transmit only one codeword. But considering the cost of transmitting the huge codebook, the two methods are virtually identical. The best codebook (set of units) is usually somewhere in the middle.

2.1. Entropy of the Source Text: $-\log L(O|CB)$

An exact calculation of source entropy is impossible, because some higher level structures such as phrasal units and grammar are not easily computable. But we can approximate source entropy using the familiar N-gram model, assuming a Markovian source. Here we find unigram model to be both powerful enough and computationally manageable. Under the unigram model (which is assumed throughout the paper), the entropy of source text is:

$$-\log L(w_1, w_2, \cdots) = -\sum_{all\ unit\ tokens} \log p(w_i)$$

where $p(\cdot)$ is the unigram model. Here we assume $p(\cdot)$ equals the empirical distribution, i.e. we take the ML estimate on the same data.

2.2. Entropy of the Codebook: $-\log L(CB)$

Again an exact calculation of codebook entropy is not easy, so we ignore for now any sub-structures (such as morphemes in words) or regularities in the codebook. Assuming the codebook is transmitted literally, its entropy is:

$$Entropy(CB) = \sum_i Cost(unit_i) = \sum_i k * length(unit_i)$$

[1] In cases where units have salient features, a simple segmentation approach might be enough to identify them.

where k is the average entropy per surface symbol.

2.3. Empirical Evidence

Minimization of the overall entropy reveals the best codebook (set of units). While this sounds merely like a conjecture, we have empirically verified in several situations that true units do yield the lowest entropy.

Using approximations mentioned before, we computed the overall entropy on 2568 sentences of SWB text, using three different types of encoding units: characters, words and sentences. It is clear from Table 1 that:

- words yield the best coding efficiency;
- the larger the unit is, the better the likelihood of the data is, although the codebook size also increases;
- entropy of the corpus when encoded with characters is equal to entropy of the codebook when encoding with sentences, because the codebook *is* the entire corpus in the latter case.

We also measured entropy of a pronunciation dictionary using different encoding units: phonemes, syllables, and words. Similarly, syllables have the lowest entropy among the three, which shows clear evidence that syllables form an intermediate structure in pronunciation.

More experiments are conducted to further verify the criterion. We tried some sequence finding techniques to find alternative encoding units, but have not observed lower entropies than that obtained with natural units. Even experiments presented later in the paper still lag behind the performance of natural units.

3. SEARCH ALGORITHM & EXPERIMENTS

Theoretically, we could search all possible segmentations to find one that yields the minimum entropy. However, this would be computationally infeasible.

If considering only the first part of the criterion, i.e. ignoring the codebook cost, the problem is reduced to finding a model and a segmentation that yield the best likelihood. We can view the segmentation as hidden information, and use the EM (Expectation-Maximization) algorithm to search for the best model:

1. Initialize the model: for example, a model that counts all possible sequences (up to a maximal length) that appeared in the corpus;
2. With the initial model, re-segment the entire corpus, find the most probable segmentation using either Viterbi or forward-backward algorithm;
3. Re-estimate the model using the new segmentation;
4. Repeat steps 2 & 3 until certain stop criterion is met.

This iterative process is guaranteed to increase the likelihood of the data. For detailed proof as well as implementation issues, we refer the reader to [5].

But this accounts for only the likelihood part of the criterion. Below we show how to fold the codebook entropy term into the EM procedure.

Since the objective function is

$$\begin{aligned} & Entropy(Source\ text) + Entropy(Codebook) \\ = & \sum_{unit\ tokens} -\log p(w_i) + \sum_{unit\ types} Cost(u_j) \end{aligned}$$

Encoding Unit	# Unit Types	# Unit Tokens	$-\log L(Corpus)$	# Chars in Codebook	Codebook Cost	Overall Entropy
Character	39	158213	6.84e+05	39	negligible	6.84e+05
Word	3288	39956	3.35e+05	21069	0.90e+05	4.24e+05
Sentence	2568	2568	0.29e+05	158213	6.84e+05	7.13e+05

Table 1: SWB Text Entropy Using Different Units. Average entropy per character is estimated to be 4.32bits.

rewriting the second summation over all unit tokens,

$$\sum_{unit\ types} Cost(u_j) = \sum_{unit\ tokens} \frac{Cost(u_j)}{\#(Occurrences\ of\ u_j)} = \sum_{unit\ tokens} \frac{Cost(u_j)}{N * p(u_j)}$$

where N is the number of tokens in the source text, we get:

$$Overall\ Entropy = \sum_{unit\ tokens} -\log p(u_i) + \frac{Cost(u_i)}{N * p(u_i)}$$

Thus the codebook cost can be viewed as a penalty term on pure ML probabilities. We can substitute the penalized probability for $p(u_i)$ in the EM procedure, i.e. the update rule for $p(u_i)$ is now:

$$p(u_i) = \hat{p}(u_i)\, e^{-\frac{Cost(u_i)}{N * \hat{p}(u_i)}}$$

where $\hat{p}(u_i)$ is the ML estimate for $p(u_i)$.

Below we present three experiments on word induction in English (Table 2). Results are reported in precision and recall rates, defined as percentage of correctly identified words over all hypothesized and reference words, respectively.

3.1. L_0 experiment

The first experiment is set up by picking 200 words (randomly), and randomly concatenating them to produce 5000 sentences. All spaces are removed and the task is to to recover them.

As shown in Table 2, precision and recall are both over 99%. Typical mistakes are confusions between pairs of words that when concatenated, produce the same string, for example:

```
ref: FIVE LIKE TOOK NOW THESE ON CAN ACTUALLY
hyp: FIVE LIKE TOO KNOW THESE ON CAN ACTUALLY
```

Other confusion pairs include `A WAY` and `AWAY`, `GO THE` and `GOT HE`, etc. The correct segmentation usually shows up as the second best candidate. And the algorithm converges quickly. The sentence error rates after each iteration are

$$2.60\% \rightarrow 1.64\% \rightarrow 1.56\% \rightarrow 1.56\% \cdots$$

This illustrates the feasibility of unsupervised unit induction. Next, we move on to a natural language corpus.

3.2. SWB Experiment

We took 2568 sentences from the SWB corpus and removed the spaces. The task again is to identify words.

This turned out to be a much more difficult task due to Zipf's law: there are lots of singletons/doubletons in any text corpus, regardless of the size of the corpus. In SWB, singletons account for almost half of all word types. Not surprisingly, they are split up into pieces. One might be tempted to conclude that fragmentation is unavoidable, because frequency is after all the major factor in promoting a unit. However, we believe that to some extent, fragmentation is avoidable. Breaking up a sequence into pieces has its own penalty: there are more probability terms in the likelihood calculation, which normally leads to lower likelihood.

We observed that our criterion was still sound: there was a consistent gap between the lowest entropy the algorithm produced and that from the true segmentation (i.e. real words). This means that search is still a problem. Indeed, careful examining reveals that singletons are mostly pruned at early iterations. Singletons are usually quite long to incur a hefty penalty. Also the search is not error tolerant: pruning errors made at early stages are not recoverable.

So we changed our segmentation strategy to allow for the emergence of new units that were not previously in the codebook. We followed Raman's method in computing probabilities for novel words [8]. The basic idea is discounting, as commonly used in language modeling: a certain portion of the probability space is withheld for novel words. We also take into account the letter composition of the word:

$$p(w) = P_{heldout} * \prod_i p(w_i)$$

Here w_i is the ith character in word w, whose probability $p(w_i)$ is estimated from a character distribution. $P_{heldout}$, the heldout probability space, can be estimated using various discounting methods.

We also adopted the more conservative search strategy in Raman's work, which is considerably faster than ours during bootstrapping.

Precision and recall rates are around 67%. A typical segmentation looks like this:

```
AND ITSEEMS LIKE UH WE 'RE <SIL> CATCH
ING ALL THAT RESIDUE I'M NOT SURE IF IT'S
KEROSENEOR WHAT THAT'S DROPP ING
```

This looks quite appealing except that certain words are broken apart and some words are glued together. Common morphemes, such as `'RE` and `ING`, are hypothesized as individual words. This is understandable since they are also units (albeit at a different level) by themselves. We still have not reached the entropy obtained using real words, which means there is still room for further improvements in the search algorithm.

3.3. Word Induction on CHILDES corpus

We also tested our algorithm on some phonemic transcripts of spontaneous speech by parents to young children. The original transcripts were made by Bernstein-Ratner[3] from the CHILDES collection [7]. The version we used was provided by M. Brent, who changed the transcription to ensure every occurrence of the same word was transcribed identically. Here are some sample utterances, their orthographic equivalents and segmentation results:

ref: yu want tu si D6 bUk	you want to see the book
hyp: yu want tu si D6 bUk	
ref: lUk D*z 6 b7 wIT hIz h&t	look there's a boy with his hat
hyp: lUk D* z 6b7 wIT hIzh &t	

Some of the errors are understandable, such as `D*z`(there's) ⇒ `D* z`(there 's), because `'s` is a frequent morpheme. Errors like `hIz h&t`(his hat) ⇒ `hIzh &t`(hish at) can be alleviated if

Corpus	# Utterances	# Words	Entropy with Real Units	Entropy Achieved by Algorithm	Precision (%)	Recall (%)
L_0	5000	98353	7.55e+05	7.55e+05	99.9	99.8
SWB	2568	39956	4.24e+05	4.38e+05	67.4	66.5
CHILDES	9790	33397	2.89e+05	2.98e+05	72.2	72.4

Table 2: Results of Word Induction Experiments

we use a more powerful morphological model for the codebook cost estimation. For example, a position dependent phoneme distribution would assign a very low probability to words ended in `h` such as `hIzh`.

Precision and recall rates are around 72%, better than results previously reported. (But note here we allow multiple passes through the corpus, which is different from the one-pass incremental condition used in [4] and [8].) This task is in essence not any different from the orthographic word induction task, except that words here are written in phonemes instead of letters. It is slightly easier because about 21% of all the utterances have only a single word. However, singletons still amount to 32% of all word types, like in any other natural language corpus.

4. RELATED WORK

Deligne proposed a multigram framework for inference of variable length linguistic and acoustic units, also using the MDL principle and EM algorithm [5]. This was first applied to an English text segmentation problem. However, the performance was not measured. For model size control, sequences of low frequency were pruned. Deligne then applied the model to find acoustic subword units from speech data, and conducted speech recognition using these units. Although the recognition accuracy is not as good as conventional systems, the experiments are interesting by themselves.

Brent also proposed an MDL framework for segmentation and word discovery in child directed speech [4]. His model is further simplified and improved by Raman [8]. Coming from a language acquisition perspective, they are more interested in an incremental lexicon building process, rather than unsupervised word induction. Their search algorithm is quite unique. It seems to be specifically tailored to the CHILDES corpus to take advantage of single-word utterances, but it works quite well on the SWB task too. However, it performs poorly on the L_0 task, suffering from L_0 corpus's lack of short sentences.

Except for estimating probabilities of novel words, our work is done independently of both works mentioned above. The main contribution of this paper is that we established a directly computable induction criterion, and empirically showed that minimization of the criterion does lead to true units. We developed a novel algorithm that directly optimizes this criterion and showed that word induction is solvable.

For Japanese word segmentation, some interesting unsupervised methods are recently proposed. Tomokiyo and Ries tried to find a segmentation of Japanese text which minimizes the language model perplexity, or equivalently, maximizes the likelihood of the test corpus. We note that this is an approximation of the MDL criterion when the number of merges is small (therefore the codebook cost is negligible). Ando and Lee used character N-gram counts, a type of local statistics, to measure the cohesion of a sequence and predict the possiblility of a boundary in Japanese kanji text [1]. The performance compares favorably to that of rule-based systems. But as a threshold-based method, careful tuning of system parameters is inevitable.

5. DISCUSSIONS & FUTURE WORK

The source-coding framework is very general and powerful. Even with very crude codebook cost estimation and likelihood estimation using a unigram model, impressive results have been achieved. It would be interesting to see how much we can improve by using more sophisticated models, such as bigram/trigram models and morphological knowledge. It would also be interesting to apply the framework to more practical applications such as phrase-based language modeling, segmentation of agglutinative language and syllabification of phonetic transcription. Although each task has its own singularity, we believe the essence of unit induction is well captured in our model.

6. ACKNOWLEDGEMENTS

The author would like to thank Michael Brent for providing his version of the CHILDES corpus, Anand Raman, Xiaojin Zhu for insightful discussions, Klaus Zechner, Lian Keng Lim for proofreading the paper, and other members of ISL for their support.

7. REFERENCES

[1] R. Ando and L. Lee. Unsupervised statistical segmentation of japanese kanji strings. Technical Report CS TR99-1756, Cornell University, 1999.

[2] M. Bacchiani and M. Ostendorf. Using automatic-derived acoustic sub-word units in large vocabulary speech recognition. In *ICSLP98*, 1998.

[3] Bernstein-Ratner. The phonology of parent child speech. *Children's Language*, 1987.

[4] M. Brent. An efficient probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 1999.

[5] S. Deligne and F. Bimbot. Inference of variable-length linguistic and acoustic units by multigrams. *Free Speech Journal*, 1997.

[6] T. Holter and T. Svendsen. Combined optimisation of baseforms and model parameters in speech recognition based on acoustic subword units. In *Proc. IEEE ASRU Workshop*, 1997.

[7] B. MacWhinney and C. Snow. The child language data exchange system. *Journal of Child Language*, 1985.

[8] A. Raman. Map lexicon is useful for segmentation and word discovery in child directed speech. unpublished, 1999.

[9] L. Tomokiyo and K. Ries. What's in a word: Learning base units in japanese for speech recognition. In *Proceedings of the ACL Workshop on Natural Language Learning*, 1997.