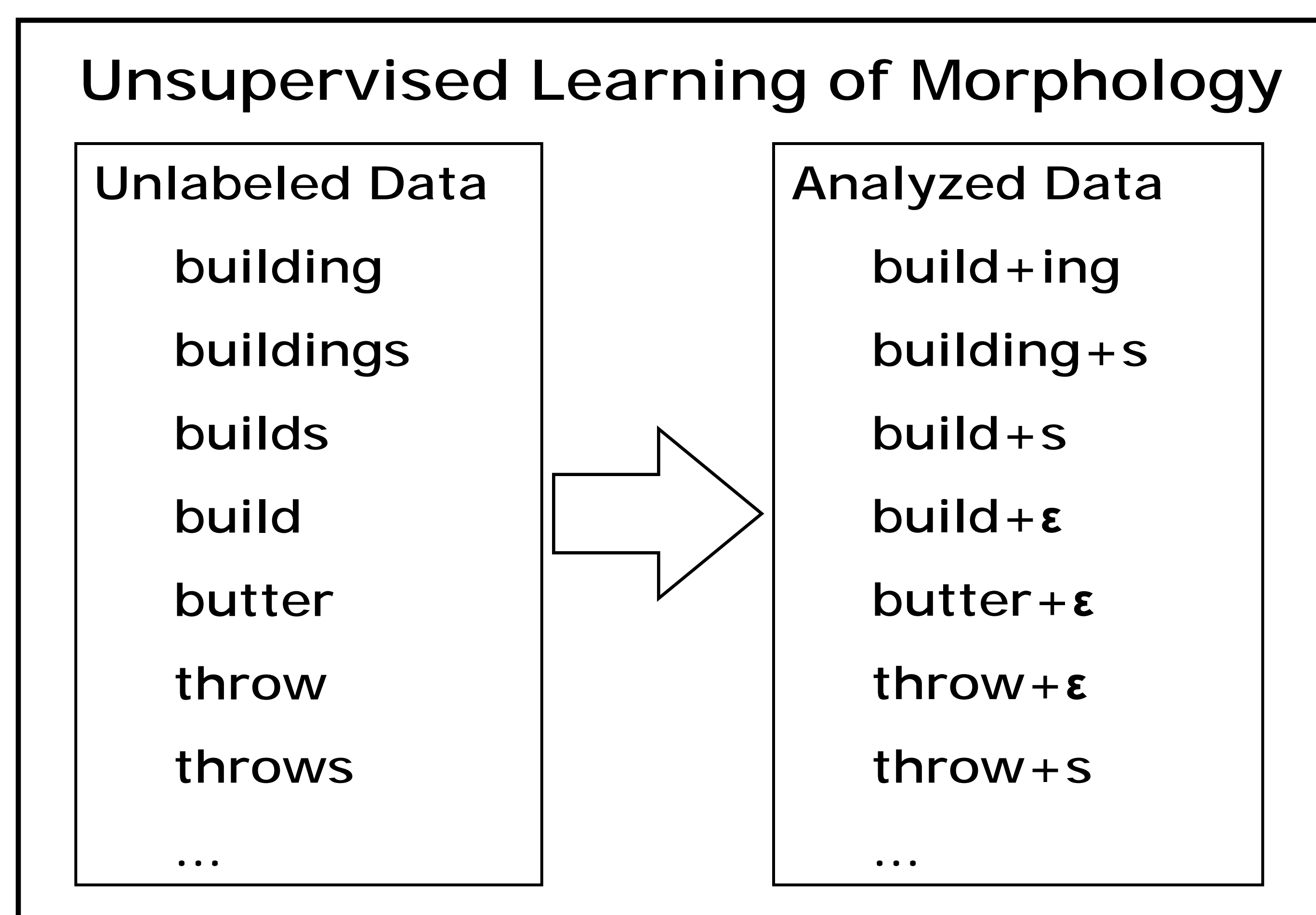


A Probabilistic Model for Learning Concatenative Morphology

Introduction

One of the fundamental problems in computational linguistics is adaptation of language processing systems to new languages with minimal reliance on human expertise. An ubiquitous component of language processing systems is the morphological analyzer, which determines the properties of morphologically complex words like *watches* and *gladly* by inferring their derivation as *watch+s* and *glad+ly*. The derivation reveals much about the word, such as the fact that *glad+ly* share syntactic properties with *quick+ly* and semantic properties with its stem *glad*. While morphological processes can take many forms, the most common are suffixation and prefixation (collectively, *concatenative morphology*).

In this work, we present a system for unsupervised inference of morphological derivations of written words, with no prior knowledge of the language in question. Specifically, neither the stems nor the suffixes of the language are given in advance. This system is designed for concatenative morphology, and the experiments presented focus on suffixation. It is applicable to any language for which written words lists are available. In languages that have been a focus of research in computational linguistics, the practical applications are limited, but in languages like Polish, automated analysis of unannotated text corpora has potential applications for information retrieval and other language processing systems. In addition, automated analysis might find application as a hypothesis-generating tool for linguists or as a cognitive model of language acquisition. In this work, however, we focus on the problem of unsupervised morphological inference for its inherent interest.



Previous Work

During the last decade several minimally supervised and unsupervised algorithms have been developed. Gaussier describes an explicitly probabilistic system that is based primarily on spellings. It is an unsupervised algorithm, but requires the tweaking of parameters to tune it to the target language. Brent and Brent et al. describe Minimum Description Length, (MDL), systems. Goldsmith describes a similar MDL approach. Our motivation in developing a new system was to improve performance and to have a model cast in an explicitly probabilistic framework. We are particularly interested in developing automated morphological analysis as a first stage of a larger grammatical inference system, and hence we favor a conservative analysis that identifies primarily productive morphological processes (those that can be applied to new words).

Our Goals

In this work, we present a probabilistic model and search algorithm for automated analysis of suffixation, along with experiments comparing our system to that of Goldsmith. This system, which extends the system of Snover and Brent, is designed to detect the final stem and suffix break of each word given a list of words.

- Only attempts to detect the final stem and suffix of each word.
- It does not distinguish between derivational and inflectional suffixation.
- It does not distinguish between the notion of a stem and a root.
- It does not currently have a mechanism to deal with multiple interpretations of a word, or to deal with morphological ambiguity.
- Within its design limitations, however, it is both mathematically clean and effective.

The Paramorph System

Our system, which we shall refer to as the Paramorph System consists of three components.

1. **A prior probability distribution** on the space of all morphological hypotheses which is used to score each hypothesis found by the system. It is this probability that the system tries to optimize.
2. **A directed search** which identifies highly productive groups of suffixes and attempts to apply them to as many stems as possible. The algorithm uses this method to find an initial hypothesis which it then refines.
3. **A hill climbing search** which attempts to further optimize the hypothesis found by the directed search. This search will move stems from one paradigm (a group of suffixes) to another similar paradigm, in order to increase the probability of the hypothesis.

Prior Probability Distribution

This section introduces a prior probability distribution over the space of all hypotheses, where a hypothesis is a set of words, each with morphological split separating the stem and suffix. The distribution is based on a seven-step model for the generation of hypotheses, which is heavily based upon the probability model presented in Snover and Brent 2001. The hypothesis is generated by choosing the number of stems and suffixes, the spellings of those stems and suffixes, and the combination of the stems and suffixes.

The seven steps are presented below, along with their probability distributions and a running example of how a hypothesis could be generated by this process. By taking the product over the distributions from all of the steps of the generative process, one can calculate the prior probability for any given hypothesis. What is described in this section is a mathematical model and not an algorithm intended to be run.

1. **Pick the number of stems, M , and suffixes, X** , using an inverse square distribution. This distribution is relatively uniform and ranges from 1 to infinity. [$M = 5, X = 3$]
2. **Pick the length of each stem and suffix**, also using an inverse square distribution. Note that suffixes of length zero are allowed, whereas all stems must be of length one or greater. [*Stem Lengths* = 4, 4, 4, 3, 3. *Suffix Lengths* = 2, 0, 1]
3. **Pick the letters in the Stem and Suffix sets**. This probability is simply the product of the probability of picking each letter. The probability of picking each character is estimated using a maximum likelihood distribution. [*Stems* = "walk", "look", "door", "far", "cat". *Suffixes* = "ed", ε, "s"]
4. **Pick the number of Paradigms, P** . A paradigm is a set of suffixes and the set of stems which attach to that exact set of suffixes. Each paradigm must have at least one stem in it, thus the number of paradigms can range from 1 to M , so a uniform probability distribution can be used. [$P = 3$]
5. **Pick the number of suffixes each paradigm represents**, using a uniform probability distribution from 1 to X . [*Number of suffixes in each paradigm* = 2, 1, 2]
6. **Pick the suffixes in each paradigm**. Since there are a finite set of suffixes of a given size, a uniform probability is assigned to each such set. [*Suffix sets* = { "ed", ε, "s" }, { ε }, { ε, "s" }]
7. **Pick a paradigm for every stem**. The probability of a stem being assigned to a paradigm is determined using a maximum likelihood estimate, such that the probability is proportional to the number of stems in that paradigm. [*look* → { "ed", ε, "s" }, *walk* → { "ed", ε, "s" }, *far* → { ε }, *door* → { ε, "s" }, *cat* → { ε, "s" }]

Posterior Probability Distribution

Typically one wishes to use a posterior probability distribution, rather than a prior probability distribution. However our prior probability distribution is proportional to the posterior probability distribution. The following equation shows how one might calculate the posterior probability of a hypothesis.

$$\Pr(\text{Hyp} | \text{Data}) = \frac{\Pr(\text{Data} | \text{Hyp}) \Pr(\text{Hyp})}{\Pr(\text{Data})}$$

Because our search algorithms only consider hypotheses that are consistent with the input lexicon and our hypotheses completely specify the input lexicon, the $\Pr(\text{Data} | \text{Hypothesis}) = 1$. The data is the same across all hypotheses considered, so the $\Pr(\text{Data})$ is simply a constant. Thus the posterior probability is equal to the prior probability divided by a constant value.

Directed Search

The Directed Search is accomplished in two steps.

1. **Sub-hypotheses, each of which is a hypothesis about a subset of the lexicon, are examined and ranked.** Each sub-hypothesis corresponds with a subset of the set of possible suffixes. The maximal set of stems is then assigned to that set of suffixes, such that for every combination of stem and suffix in the two sets, the resulting word is in the input lexicon.

The score of a sub-hypothesis is the probability of the words in the hypothesis with the assigned breaks, divided by the probability of all those words as stems attached to the null suffix.

2. **The 100 best sub-hypotheses are then incrementally combined until a single sub-hypothesis remains.** The remainder of the input lexicon is added to this sub-hypothesis at which point it becomes the final hypothesis, which is passed onto the Hill Climbing Search.

The sub-hypotheses are combined by repeatedly selecting the highest ranked sub-hypothesis, and adding it into the other sub-hypotheses. The breaks in the higher ranking sub-hypothesis override any previous breaks in the new sub-hypotheses. The sub-hypotheses then need to be rescored. Those sub-hypotheses which shared stems and suffixes with the highest ranking sub-hypothesis will have their scores increased, whereas other sub-hypotheses may have lower scores.

The combining of sub-hypotheses continues until all remaining sub-hypotheses have scores less than 1, or all sub-hypotheses have been merged. At this point the final sub-hypothesis that had score greater than 1 has all remaining words added into it, with the null suffix, resulting in a hypothesis over all of the input words.

Hill Climbing Search

The Hill Climbing Search further optimizes the probability of the hypothesis found by the directed search, by attempting to move stems from paradigms to similar paradigms. The search continues to move stems until it can no longer increase the probability of the hypothesis.

The following is pseudo code for the hill climbing search.

```

While the score increases (1)
  For every possible suffix, x (2)
    For every paradigm, p (3)
      p' = the paradigm with the suffixes of p + x. (4)
      Attempt to move all stems in p to p', if it increases the score. (5)
      If the score increases Goto 2 (6)
  For every paradigm, p (7)
    For every suffix, x, in p (8)
      p' = the paradigm with the suffixes of p - x. (9)
      Move all stems in p to p', if it increases the score. (10)
      If the score increases Goto 7 (11)

```

Note that only hypotheses consistent with the input lexicon are considered. Stems are never moved to a paradigm unless the stem combined with each of the suffixes of the paradigm are words in the input lexicon.

Results

Paramorph was compared to Goldsmith's MDL system, Linguistica, on a lexicons of 500, 1000, 2000, 4000, and 8000 words in both English and Polish. The lexicons were formed from the most common words in the input corpora.

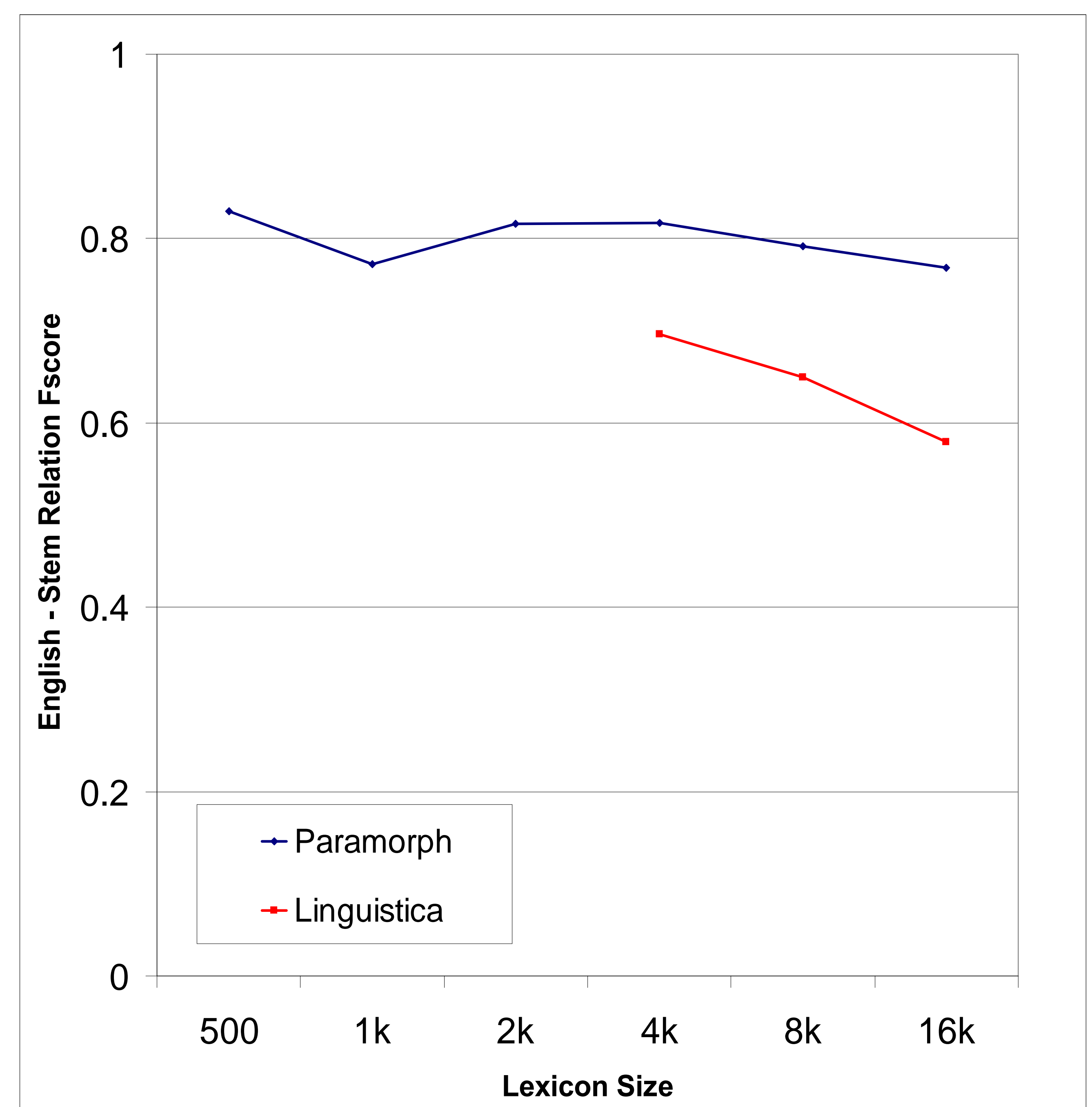
The results of the system were evaluated using a Stem Relation method. The results were measured by what percentage of words were correctly identified as having the same immediate stem, regardless of what the stem chosen was. For example "building" and "buildings" should have the same stems, as should "build" and "building". Whereas "buildings" and "build" should have different stems.

The scores shown are for Fscore, which is an unbiased combination of precision and recall. Precision is a measure of how many of the relations predicted are correct. Recall is a measure of how many of the relations that existed were predicted.

It should be noted that Linguistica has been optimized for run-time performance whereas Paramorph is still a development tool. Thus Linguistica runs much faster than Paramorph on all inputs.

The results show a consistently superior performance by Paramorph. This is due largely to the higher precision of Paramorph., resulting in highly conservative morphological analyses.

English Results 500 – 16,000 Word Types



Polish Results 500 – 8,000 Word Types

