Peter Venable (pvenable@cs)

11−751 Speech Recognition and Understanding
Class Project

12/2/98

Predicting the Next Word in Reading Tutor Stories

Advisor: Dr. Jack Mostow (mostow@lti)

Motivation

Project LISTEN is a reading tutor which listens to children as they read stories from the screen. The Tutor program also allows children to type in their own stories. The long term goal of this project is to allow it to aid these budding authors (with poor spelling) by guessing which word a child is trying to write. This is possible through language modeling. The question is, can a language model guess accurately enough to actually be useful, to be accepted by the user? The direct goal of this project is to assess how well a language model would do at guessing which word to write next in a story, given:

– the immediately preceding words
– the first k letters of the word

Language Model Training

The procedure is to build a basic language model from the stories given, and then evaluate that model in various situations. Evaluation should show how well it constrains the next word. An intuitive measure of the quality of word predictions is the number of guess words which must be listed at each point in order for the actual word to be included in the list, that is, if the correct word is the m^{th} most probable word according to the model, then it would take at least m guesses to find the correct word. However, this has the problem that it will be extremely high or infinite for hard–to–predict or unpredictable words. So instead, we can measure the empirical probability of the actual next word being one of the top m guesses, for certain values of m.

Some of the stories available for language model training are classified by grade level, from grade one (labeled A) to grade five (labeled E), but the majority of the stories are not classified. These unclassified stories are mostly of higher reading level than the graded material. Ten percent of the data from each classification is held out for testing.

The CMU–Cambridge Language Modeling Toolkit is used to build a trigram language model with Good–Turing discounting. Before the model is built, using a script like the one shown here in the box, the text is preprocessed to spell out numbers, fix up abbreviations, and remove punctuation. Case distinctions are retained, though as an afterthought it would probably have been better to have removed them.

```
# build language model gs.binlm from text lm/gs.text
text2wfreq < lm/gs.txt > gs.wfreq
wfreq2vocab < gs.wfreq > gs.vocab
text2idngram -vocab gs.vocab -gzip < lm/gs.txt > gs.idngram
idngram2stats -n3 < gs.idngram >& gs.stats
idngram2lm -idngram gs.idngram -vocab gs.vocab -binary gs.binlm
```

Two separate language models are trained, one based on all the data (except the 10% held out for testing) and one based only on the (much smaller) set of stories which are classified by grade level.

Word Prediction Method

The program to predict the next word in a story is built directly on top of the Language Modeling Toolkit. To predict each word, it loops though all possible words (as constrained by the given first k letters, where $0 \leq k \leq 2$) and uses the Language Modeling Toolkit's built in procedure to compute the probability of that word given the last two words. Then statistics are kept as to whether the correct next word was among the top m guesses. Thus the program goes through the test text in order, attempting to predict each word. For comparison, statistics are also kept of how often the correct word would have been in the top m guesses given the first *k* letters alone, without the language model, and the scores are broken down between common words and other words. The common words are detected irrespective of case, and are given by the list in the box to the right.

```
a, all, an, and, are, as, at, be, by,
for, he, her, him, his, i, if, in, is,
it, its, me, not, of, off, on, or,
she, so, the, them, then, they, this,
to, we, you
```

Evaluation Methodology

As mentioned above, the test data is 10% of the training data held aside from training.

Tests are run separately according to the following axes of variability:

1. Which language model was used: GS=graded stories only, AS=all stories.
   Note: later on in the project a problem was found with redundant data in the AS training set and test set, so only the GS−trained language was used.
2. Which test set was used: A,B,C,D,E,GS,AS
3. How many letters of the word to be guessed are given: k=0, k=1, k=2

Also, within each test run (each combination of the above factors) the separate data are collected according to the following axes of variability:

1. How many guesses are allowed: m={1,2,4,8,16}
2. Whether the language model is used, or just the first *k* letters.
3. Whether the word being guessed is on the list of common words.

So, the total number of result data is 1*7*3*5*2*2=420.

Results

A summary of the results is printed in numerical and graphical form on the following pages. As shown on page 6, fairly good predictions can be made when the first two letters of the word are known in advance, getting the correct word on the first guess 50% of the time, and within 16 guesses 80% of the time, when tested on the graded set. As the number of guesses allowed grows, the importance of knowing the first *k* letters increases, becoming fairly significant for $k \geq 2$ and $m \geq 8$. Even so, it is clear that the language model is doing much better than the *k* −constraint alone at all points. Of course, if *k* were increase too much, it would predict perfectly, obviating the language model, but that is beside the point since if we already know the word we don't need to predict it.

Moving on to the tests where k=1, we see that *k* values less than two have a negligible effect by themselves, though comparison with the k=0 tests indicates that this first letter is still quite significant when combined with the language model. That is, it has a huge impact on the combined result even though it accomplishes almost nothing by itself. On both these tests we can observe that the proportion of common words (marked F) to other words (~F) correctly predicted tends to range from about 4:1 to 1:1. This is not too surprising, since common words are naturally easier to predict than less common words. The question is, how useful is it to predict easy words? They are also the easiest words for the user to get correct without the help of the predictor. If easy words are ignored, the best results fall to about 50% for k=2, 35% for k=1, and 10% for k=0; and these are for the rather large number of guesses, m=16. So knowing even one letter of the word offers a significant advantage.

The other direction of comparison is between stories of different grade levels (page 7). However, the data from each grade level was too sparse to draw any definite conclusions about any correlation between grade level and the difficulty of predicting the next word. Intuition supposes that higher grade levels would use a larger vocabulary, have more perplexity, and therefore be harder to predict. Because of sparsity, the results are too random to verify this intuition, showing no definite trends of any kind.

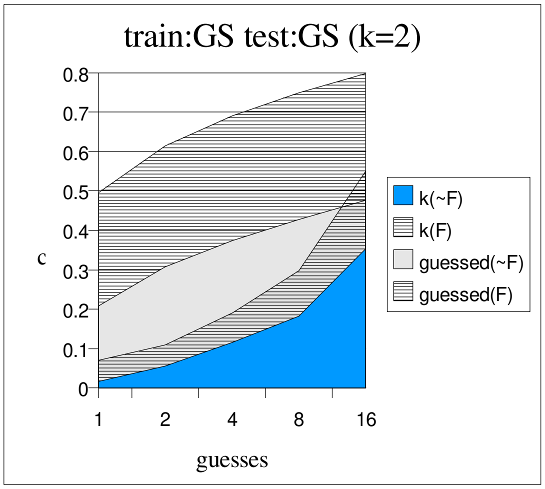
Conclusions

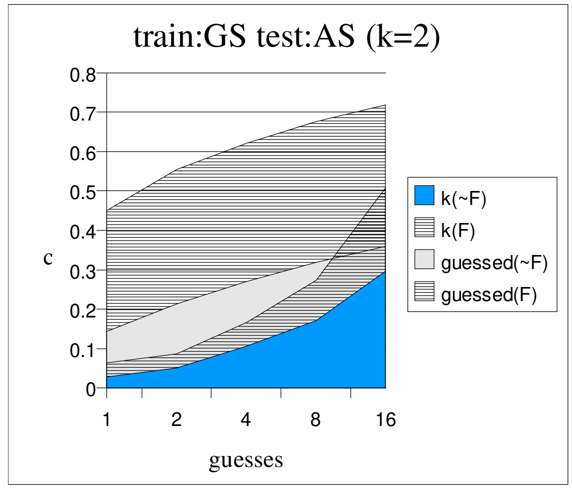
The direct goal of this project was to assess how well a language model would do at guessing which word a child will write next in a story. The results show that the method has potential, with correctness results reaching as high as 80% under certain conditions. However, they also suggest several concerns and areas for future research. One concern is the predominance of common words among those correctly predicted, which raises the question of how useful a predictor that offers help only when help is not needed. The project has shown that a constraining influence, such as knowing the first letter of a word, can be very effective in conjunction with a language model, much more so than either in isolation. An opportunity for improvement, then, would be to use a much more sophisticated heuristic taking what has been typed in and mapping that to possible matches. For example, instead of just taking the first one or two letters, a phonetic translation could be made, or the first and last letters used, or the length of the word taken into account. This would probably result in greatly improved accuracy. Also, it is unclear what effect on performance would results in relaxing the case–sensitivity of the data, but it would produce a more realistic model regardless, and is therefore desirable. Additionally, it would be interesting to try training separate models for each grade level to see if performance is thereby improved. Of course, that would require much more data, as the current collection is far too sparse. Overall, the results of this project indicate that a language model, in conjunction with some constraining influence such as an attempted spelling of the word, could probably offer correct guesses within the first few tries a reasonable percentage of the time, and thus be a useful aid to composition for authors who have difficulty spelling.

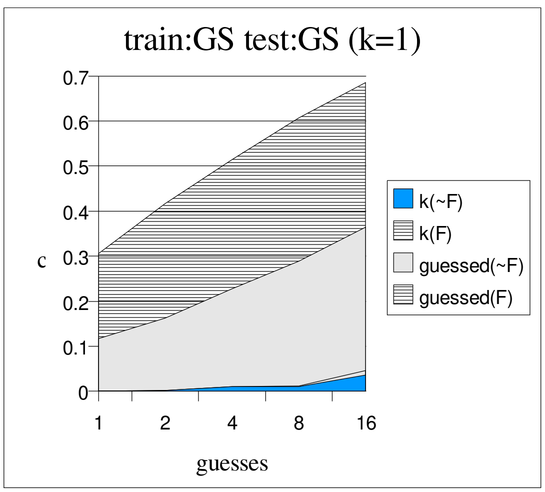
		Found by k alone.		Guessed by LM.	
Gs – gs	m	k(~F)	k(F)	G(~F)	G(F)
K=2	1	1.70%	6.96%	20.87%	49.62%
	2	5.51%	10.86%	30.70%	61.49%
	4	11.54%	19.00%	37.40%	69.04%
	8	18.32%	29.77%	42.75%	74.89%
	16	35.20%	54.96%	47.58%	79.81%
As – gs	m	k(~F)	k(F)	G(~F)	G(F)
K=2	1	2.84%	6.35%	14.32%	45.10%
	2	5.08%	8.62%	21.33%	55.35%
	4	10.66%	16.57%	27.06%	62.07%
	8	17.05%	27.38%	31.80%	67.53%
	16	29.62%	50.71%	35.95%	71.83%
Gs – gs	m	k(~F)	k(F)	G(~F)	G(F)
K=1	1	0.08%	0.08%	11.70%	30.53%
	2	0.17%	0.17%	16.20%	41.65%
	4	1.10%	1.10%	22.73%	51.48%
	8	1.10%	1.19%	28.84%	60.73%
	16	3.65%	4.58%	36.39%	68.45%
As – gs	m	k(~F)	k(F)	G(~F)	G(F)
K=1	1	0.02%	0.02%	5.76%	25.57%
	2	0.20%	0.20%	9.61%	36.16%
	4	0.33%	0.33%	14.29%	45.04%
	8	1.09%	1.13%	19.92%	54.46%
	16	3.86%	4.32%	25.78%	61.21%
Gs – gs	m	k(~F)	k(F)	G(~F)	G(F)
K=0	1	0.00%	0.00%	2.97%	9.33%
	2	0.00%	0.00%	4.33%	14.16%
	4	0.00%	0.00%	5.94%	19.08%
	8	0.00%	0.00%	6.96%	24.34%
	16	0.00%	0.00%	10.09%	31.55%
As – gs	m	k(~F)	k(F)	G(~F)	G(F)
K=0	1	0.00%	0.00%	0.77%	6.91%
	2	0.00%	0.00%	1.27%	11.60%
	4	0.00%	0.00%	1.98%	16.21%
	8	0.00%	0.00%	2.74%	23.26%
	16	0.00%	0.00%	5.08%	28.76%

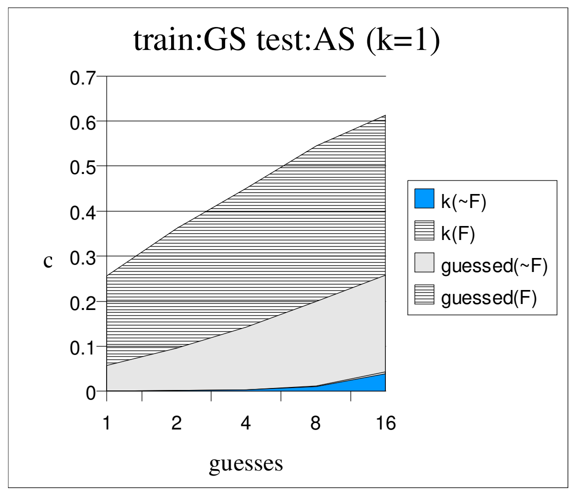
Results

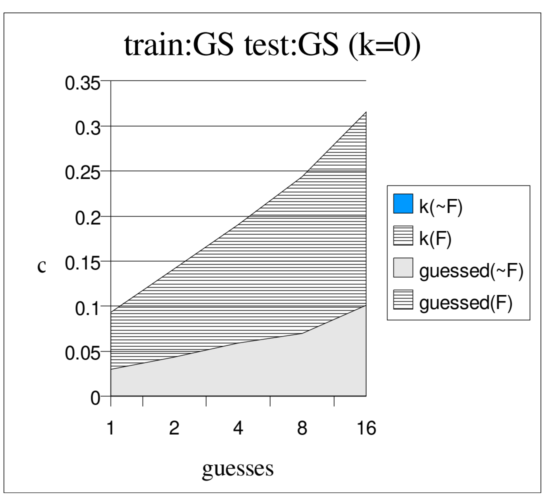
The left column tells which language model was used, which test set was used, and how many letters were given. The column marked *m* tells how many guesses were allowed. The columns marked *k* show what percentage of the words were guessed correctly by only knowing the first *k* letters, without using the language model. The columns marked *guessed* show what percentage of the words were guessed correctly by the language model. The columns marked (~F) count only words which are not on the filter list of common words, and the columns marked (F) count only words which are on the filter list.

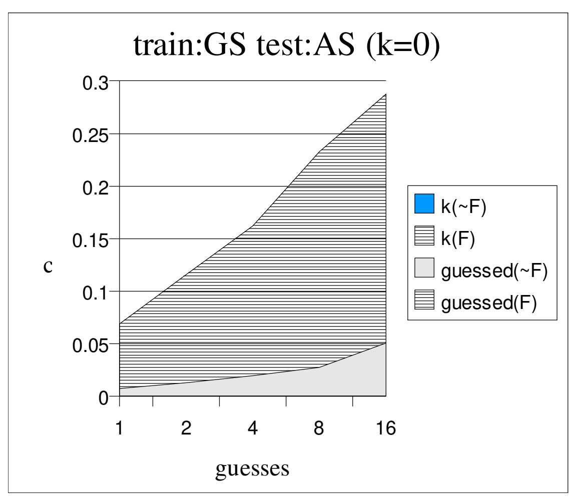

train:GS test:GS (k=2)
c
guesses
k(~F)
k(F)
guessed(~F)
guessed(F)

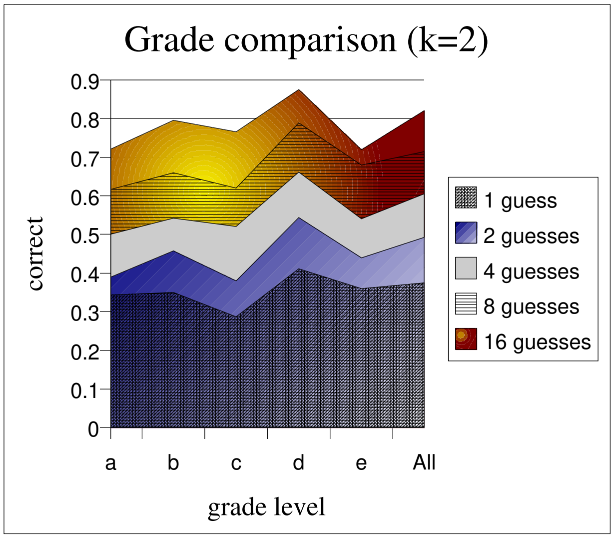

train:GS test:AS (k=2)
c
guesses
k(~F)
k(F)
guessed(~F)
guessed(F)

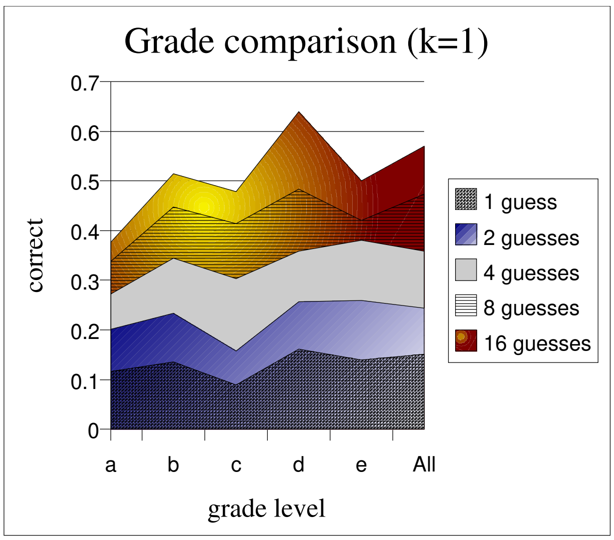

train:GS test:GS (k=1)
c
guesses
k(~F)
k(F)
guessed(~F)
guessed(F)

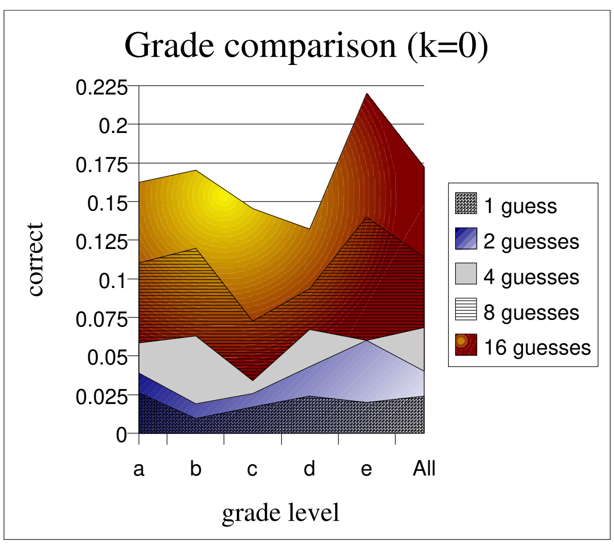

train:GS test:AS (k=1)
c
guesses
k(~F)
k(F)
guessed(~F)
guessed(F)


train:GS test:GS (k=0)
c
guesses
k(~F)
k(F)
guessed(~F)
guessed(F)


train:GS test:AS (k=0)
c
guesses
k(~F)
k(F)
guessed(~F)
guessed(F)


Grade comparison (k=2)
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
correct
a
b
c
d
e
All
grade level
1 guess
2 guesses
4 guesses
8 guesses
16 guesses


Grade comparison (k=1)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
correct
a
b
c
d
e
All
grade level
1 guess
2 guesses
4 guesses
8 guesses
16 guesses


Grade comparison (k=0)
0.225
0.2
0.175
0.15
0.125
0.1
0.075
0.05
0.025
0
correct
a
b
c
d
e
All
grade level
1 guess
2 guesses
4 guesses
8 guesses
16 guesses