On The Naturalness of Software

SID RASKAR

Computer And Information Science

University Of Delaware

Naturalness ?

- **Central Hypothesis –**
  - Natural Languages - Simple and repetitive in practice
  - Software - Natural product of human effort
  - Usefully modelled by Statistical language models
- Can be leveraged to support software engineers

Motivation

- “The European Central #### announced that interest rates remain unchanged…”
- Bank rather than fish !
- Speech Recognizer, OCR

- Similar Code Completion -
- For(i=0;i<=10
- ;i++) {

Language Model

- Assigns probability to an utterance
- Attempts to calculate maximum likelihood estimate of the parameter

N-gram Model –

- Token occurrence is influenced by the n-1 tokens that precede the token in consideration.

$$p(s) = p(a_1)p(a_2 \mid a_1)p(a_3 \mid a_1 a_2) \dots p(a_n|a_1 \dots a_{n-1})$$

What Makes a Good Model?

- Captures the regularities in the corpus, predicts tokens with high confidence
- Model will not find new document surprising
- In NLP term, cross entropy

$$H_{\mathcal{M}}(s) = -\frac{1}{n}\sum_{i=1}^{n} \log\ p_{\mathcal{M}}(a_i \mid a_1 \dots a_{i-1})$$

- Good model has low entropy
- High Probability for frequent words
- Low probability for rare words

Datasets

- Natural Language-
  - Brown and Gutenberg corpus
- For code –
  - Java projects
  - Ubuntu Applications
- Removed comments, produce token sequence
- Each project concatenated as single document

10 Fold Cross Validation

- 90% corpus for training
- 10% corpus for testing
- Unseen tokens smoothed

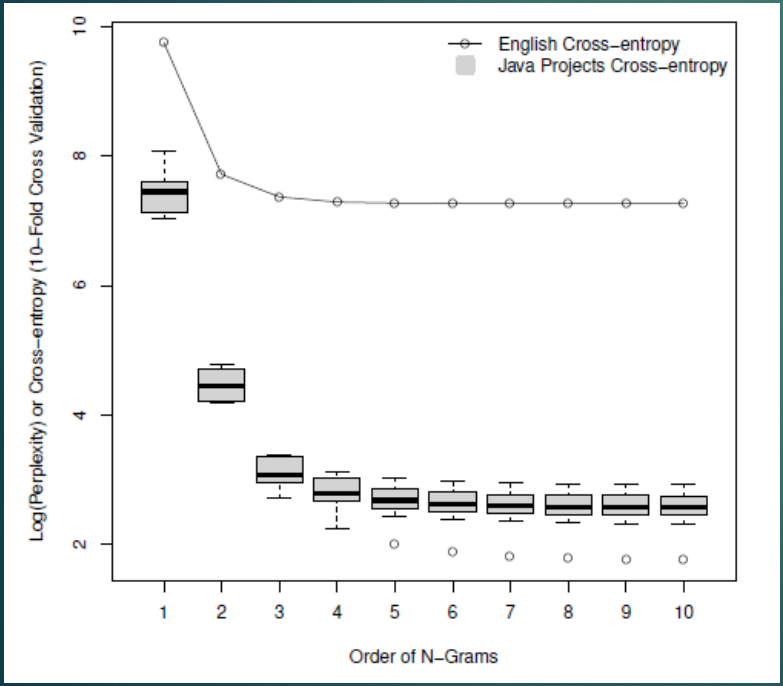
Java Project	Version	Lines	Tokens	
			Total	Unique
Ant	20110123	254457	919148	27008
Batik	20110118	367293	1384554	30298
Cassandra	20110122	135992	697498	13002
Eclipse-E4	20110426	1543206	6807301	98652
Log4J	20101119	68528	247001	8056
Lucene	20100319	429957	2130349	32676
Maven2	20101118	61622	263831	7637
Maven3	20110122	114527	462397	10839
Xalan-J	20091212	349837	1085022	39383
Xerces	20110111	257572	992623	19542

Ubuntu Domain	Version	Lines	Tokens	
			Total	Unique
Admin (116)	10.10	9092325	41208531	1140555
Doc (22)	10.10	87192	362501	15373
Graphics (21)	10.10	1422514	7453031	188792
Interp. (23)	10.10	1416361	6388351	201538
Mail (15)	10.10	1049136	4408776	137324
Net (86)	10.10	5012473	20666917	541896
Sound (26)	10.10	1698584	29310969	436377
Tex (135)	10.10	1405674	14342943	375845
Text (118)	10.10	1325700	6291804	155177
Web (31)	10.10	1743376	11361332	216474

English Corpus	Version	Lines	Tokens	
			Total	Unique
Brown	20101101	81851	1161192	56057
Gutenberg	20101101	55578	2621613	51156

“ Do n-gram language models ***capture regularities*** *in software **?** ”*

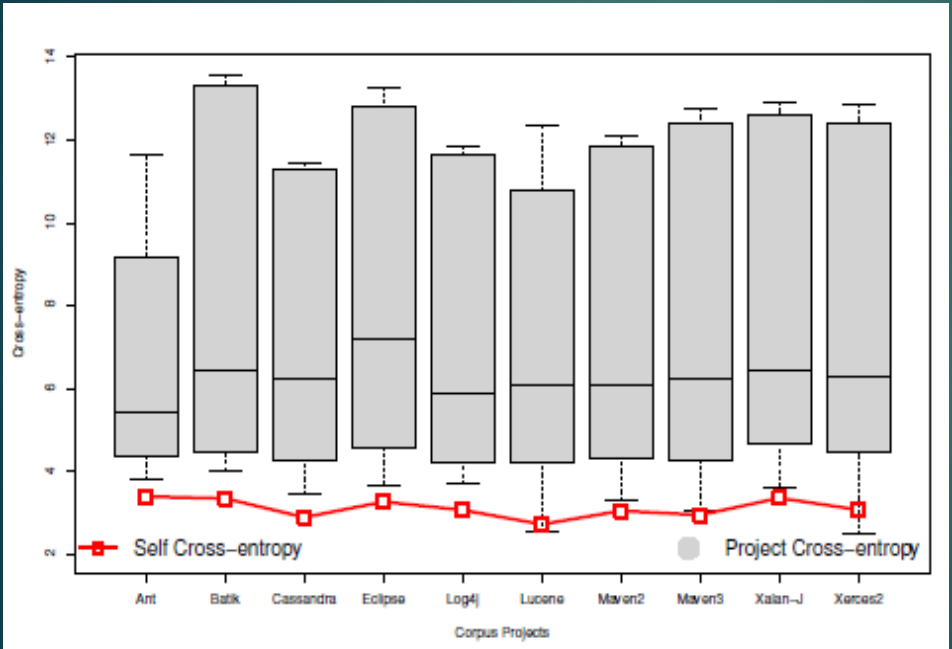
- Calculate n-gram models for English and java
- Self cross entropy



- Language model captures as much repetitive local context in Java, as it does in English
- Software is far more regular than English
- Increased similarity due to simplicity of Java?

Is the **local regularity** that the statistical language model captures merely **language specific** or is it also **project specific?**

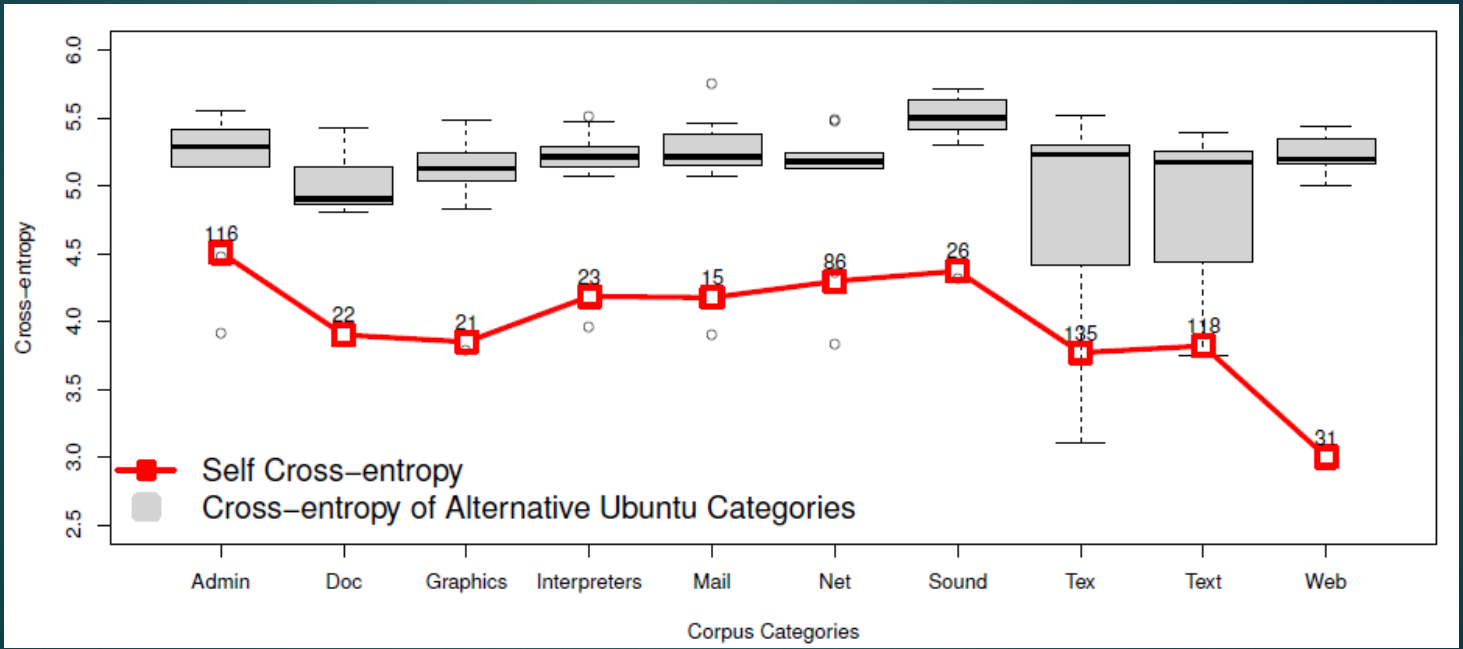
- Train model on one project and test on another to local regularity
- 10 Projects - Trigram model



- Avg Self entropy is always lower
- Useful language models can be built even for small projects.
- Captures significant levels of local regularity

Do n-gram models capture similarities within and differences between **project domain?**

- Local Regularities repeated within application domains
- Some domains have very high level of regularity eg. web



Eclipse Suggestion Plug-in

Algorithm 1 $\mathcal{MSE}$(esugg, nsugg, maxrank, minlen)

Require: esugg and nsugg are ordered sets of Eclipse and N-gram suggestions.

elong := $\{p \in \text{esugg}[1..\text{maxrank}] \mid \texttt{strlen}(p) > \text{minlen}\}$

if elong $\neq \emptyset$ **then**
  return esugg[1..maxrank]
end if
return esugg$[1..\lceil \frac{\text{maxrank}}{2} \rceil] \circ$ nsugg$[1..\lfloor \frac{\text{maxrank}}{2} \rfloor]$

Simple Merge Algorithm (*MSE*)

- Breakeven length= 7

If

ECSE offers long suggestions, pick them greedily

Else

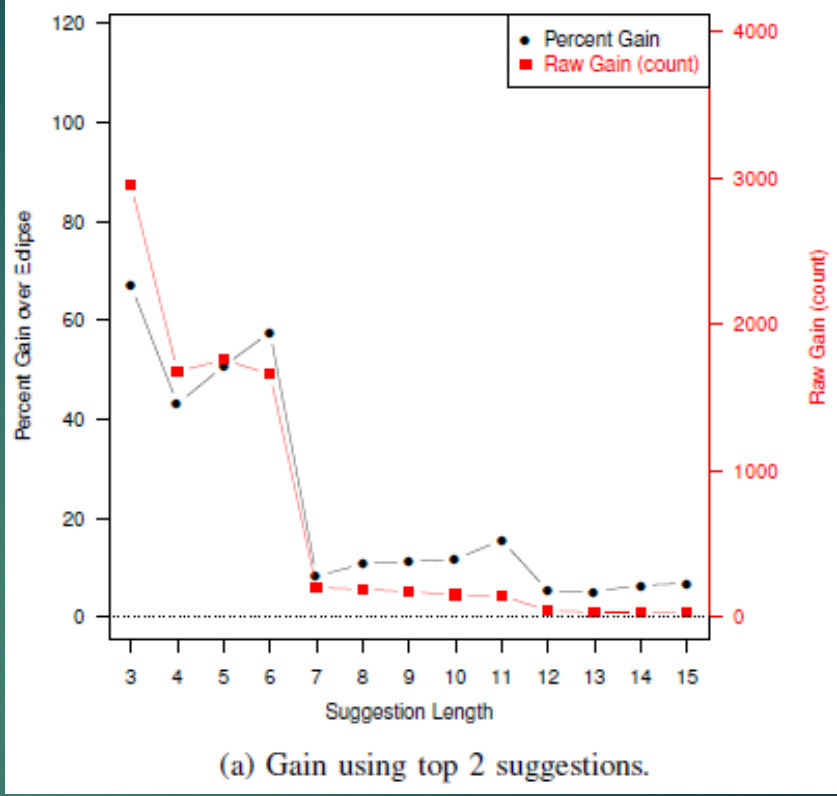
Pick half from ECSE and half from NGSE

- *NGSE* – n-gram models suggestion engine
- *ECSE* – Eclipse's built in suggestion engine

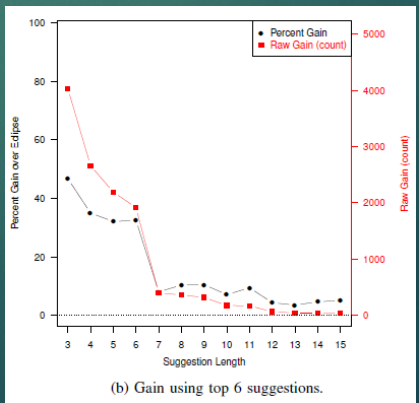
- *NGSE* –
  - Tri-gram Model
  - 0.2 seconds suggestion time

- NGSE good at recommending short tokens
- ECSE good at longer tokens

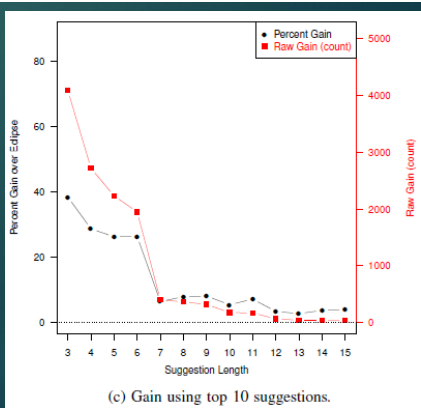
- Controlled 2 factors –
  - String length of suggestions
  - Number of choices
- Training set – 160 files
- Test set – 40 files
- Tri gram model
- MSE has advantage over ECSE – measured as the gain in number of correct suggestions.
- Gains up through 6 character tokens – 33-67%
- 7 to 15 characters – 3-16%



(a) Gain using top 2 suggestions.



(b) Gain using top 6 suggestions.



(c) Gain using top 10 suggestions.

Related and Future Work

- Naturalness of names in code
- Code Summarization
- Software Mining
- Language Models for accessibility
- Software Tools

Conclusion

Fairly simple statistical model can capture a surprising amount of regularity in natural software which can be leveraged to assist further in software development and maintenance.