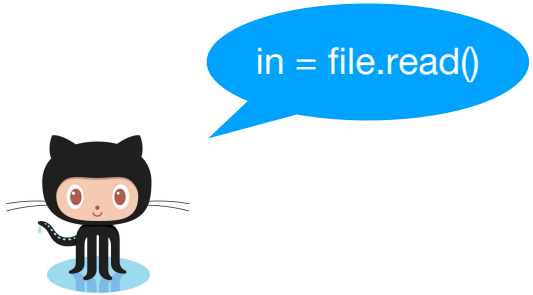
# Studying the Differences Between Natural and Programming Languages

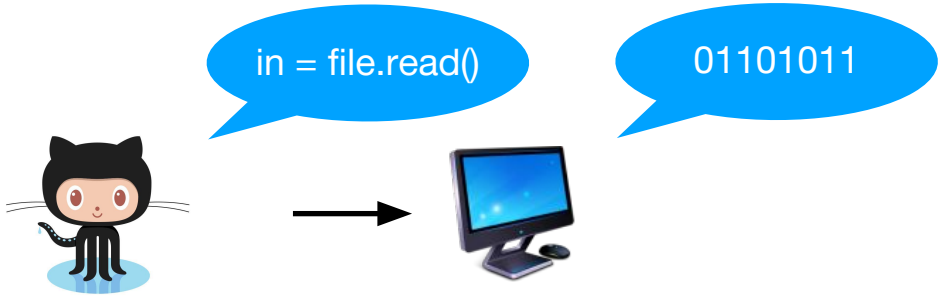
Casey Casalnuovo
Prem Devanbu


**Grant #1414172**
**Exploiting the Naturalness of Software**

# Code as Communication

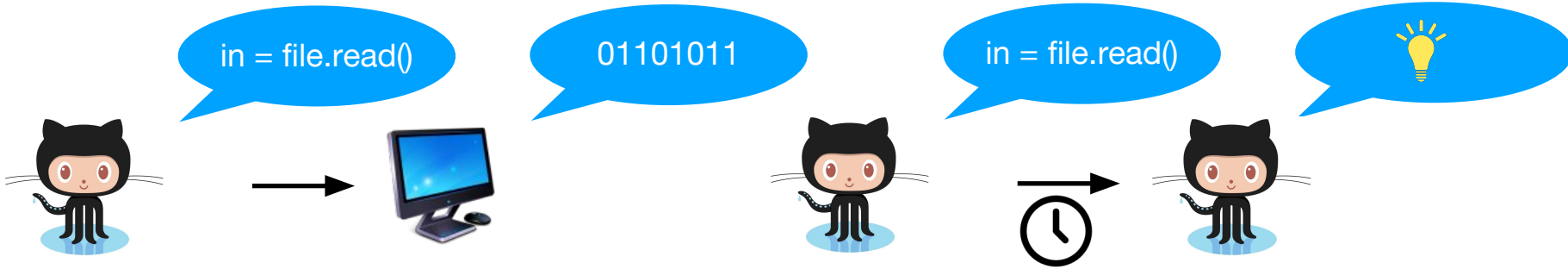


# Code as Communication

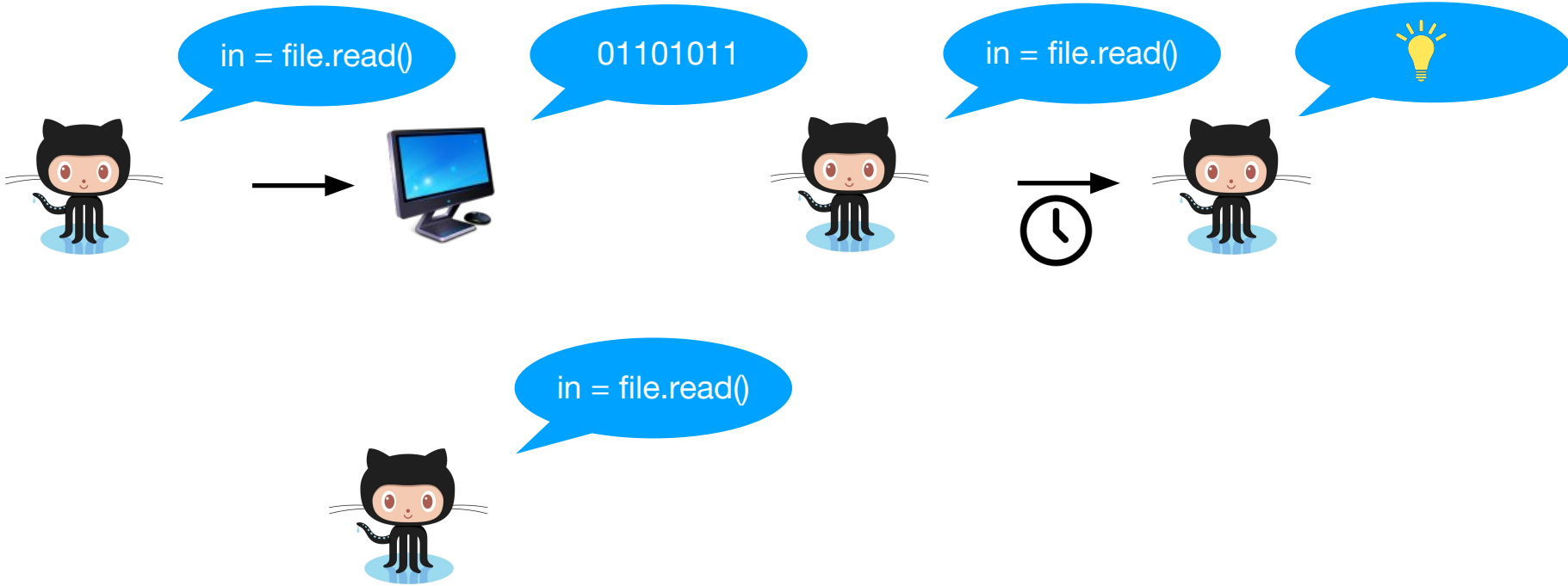


# Code as Communication

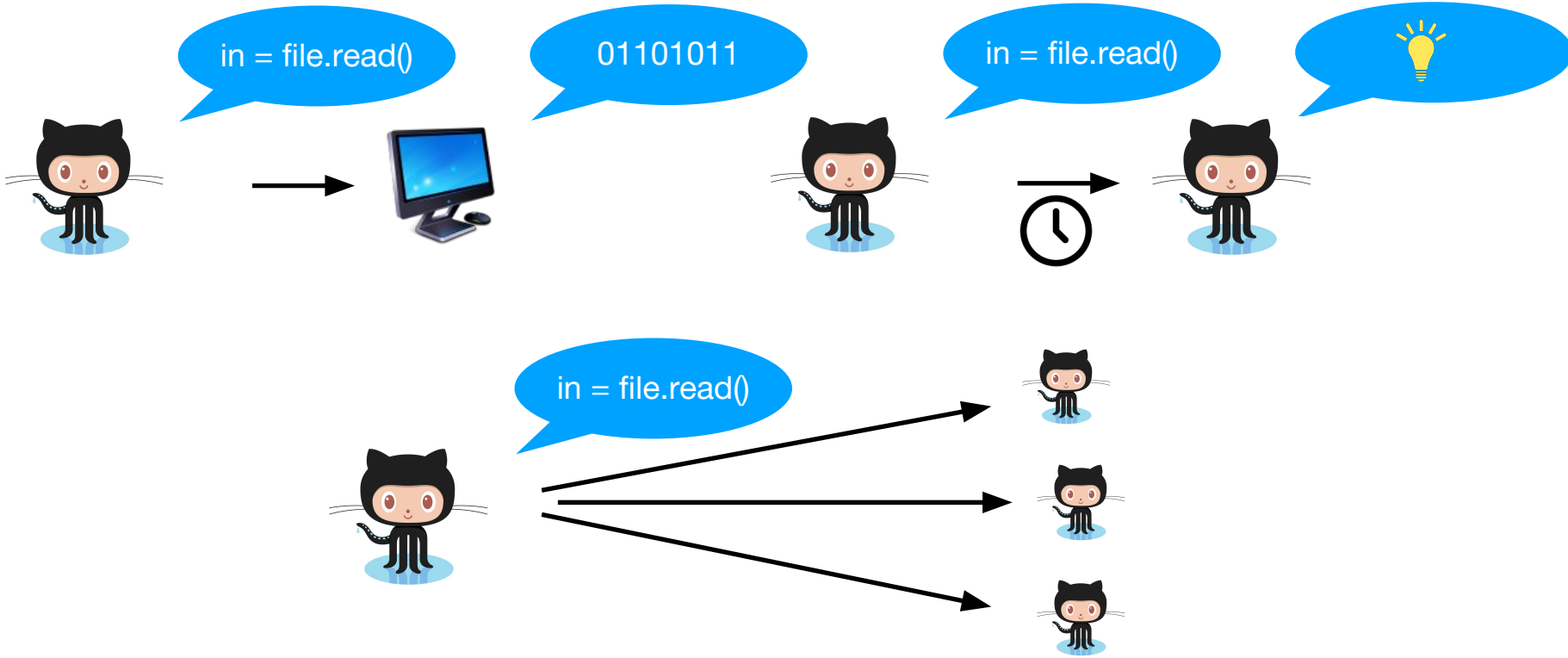
# Code as Communication



# Code as Communication



# Code as Communication

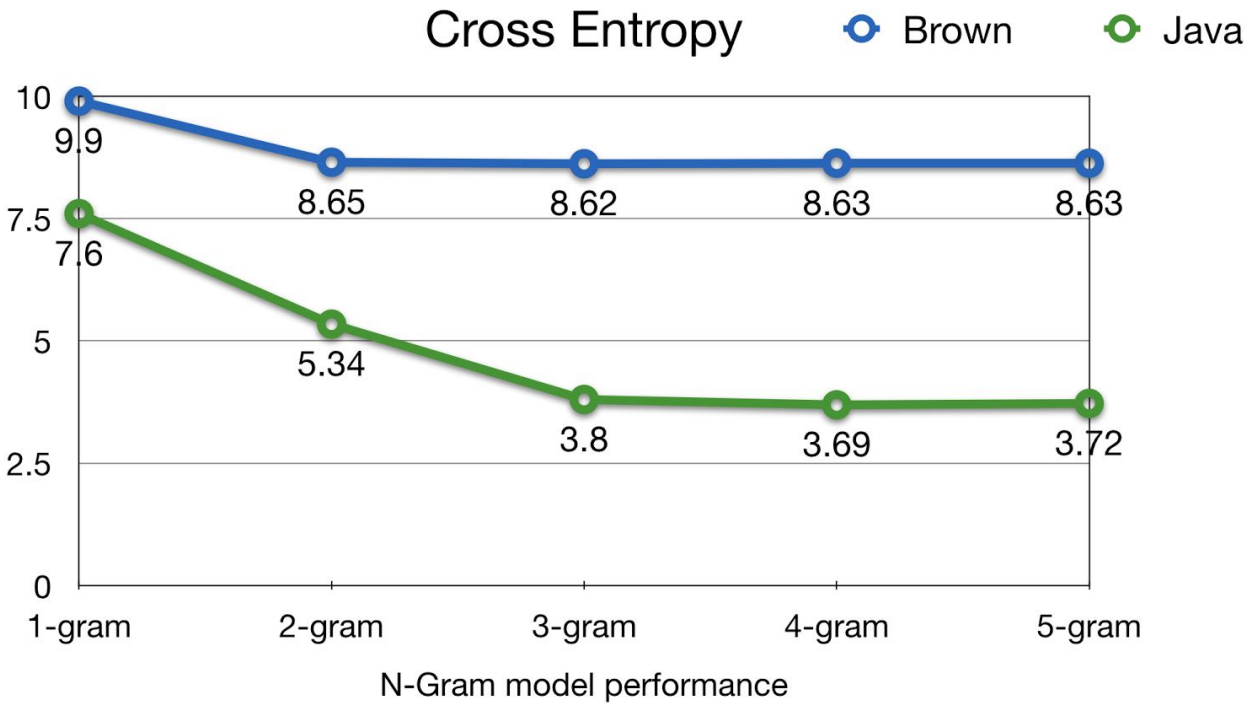


# Code as Communication

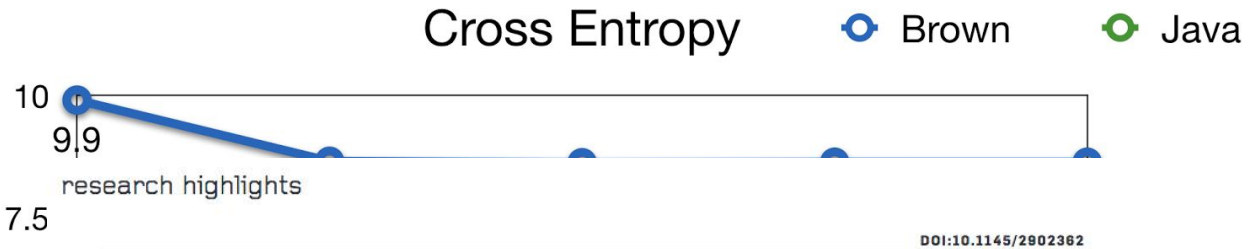

"Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do."

- Donald Knuth

# Code Is Very Repetitive



# Code Is Very Repetitive




DOI:10.1145/2902362

## On the Naturalness of Software

By Abram Hindle, Earl T. Barr, Mark Gabel, Zhendong Su, and Premkumar Devanbu

**Abstract**

**Natural languages like English are rich, complex, and powerful. The highly creative and graceful use of languages like English and Tamil, by masters like Shakespeare and Avvaiyar, can certainly delight and inspire. But in practice, given cognitive constraints and the exigencies of daily life, most human utterances are far simpler and much more repetitive and predictable. In fact, these utterances can be very usefully modeled using modern statistical methods. This fact has led to the phenomenal success of statistical approaches to speech recognition, natural language translation, question-answering, and text mining and comprehension.**

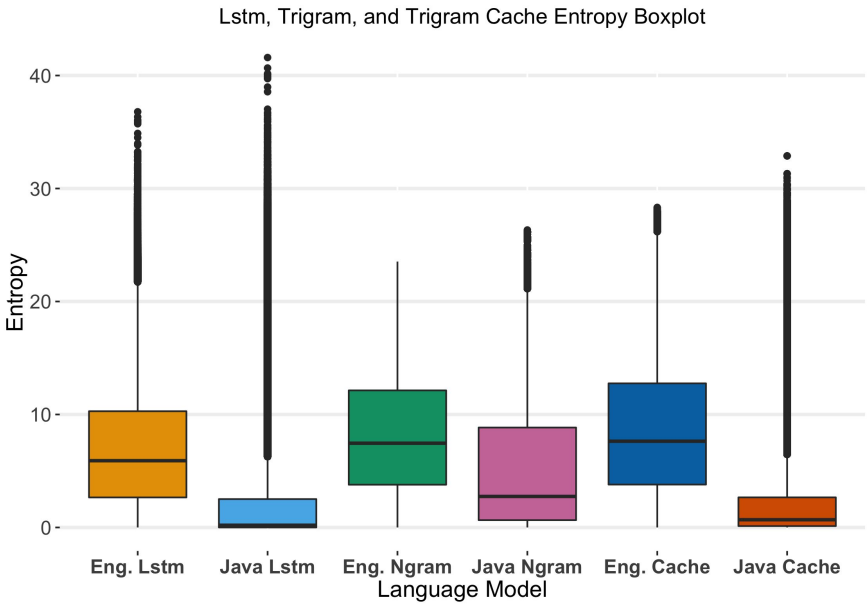
**We begin with the conjecture that *most software is also natural*, in the sense that it is created by humans at work, with all the attendant constraints and limitations—and thus, like natural language, it is also likely to be repeti-**

too cumbersome to perform practical tasks at scale. Both these approaches essentially dealt with NLP from first principles—addressing *language*, in all its rich theoretical glory, rather than examining corpora of actual *utterances*, that is, what people actually write or say. In the 1980s, a fundamental shift to *corpus-based*, *statistically rigorous* methods occurred. The availability of large, on-line corpora of natural language text, including "aligned" text with translations in multiple languages,[a] along with the computational muscle (CPU speed, primary and secondary storage) to estimate robust statistical models over very large data sets has led to stunning progress and widely available practical applications, such as statistical translation used by translate.google.com.[b]

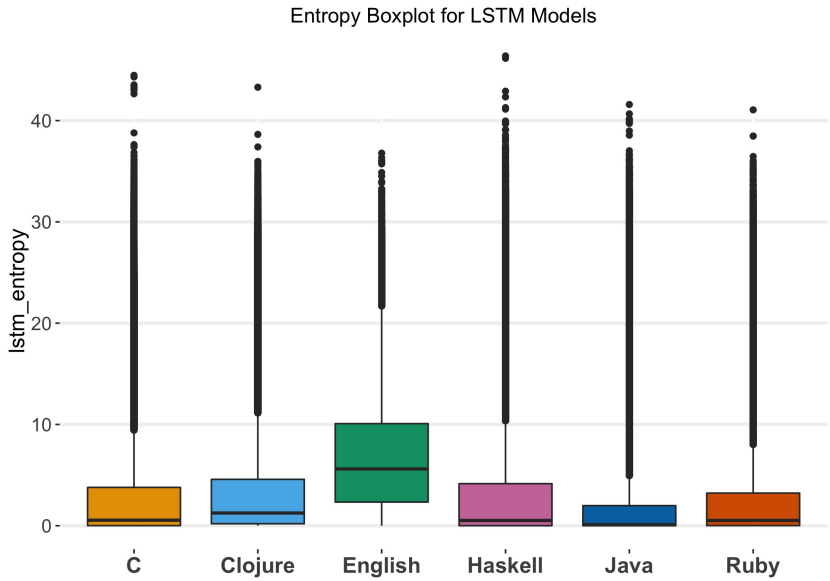
Can we apply these techniques *directly* to software, with its strange syntax, awash with punctuation, and replicate this success? The funny thing about programming is that it is as

# Code is very repetitive and predictable compared to English

**And this can be found regardless of the language model used…**

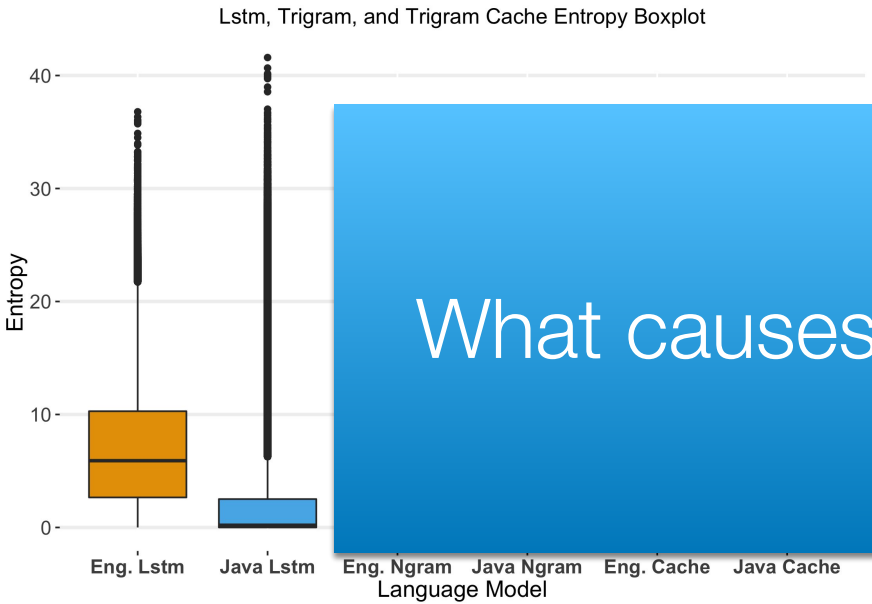



**Or regardless of the type of programming language selected…**

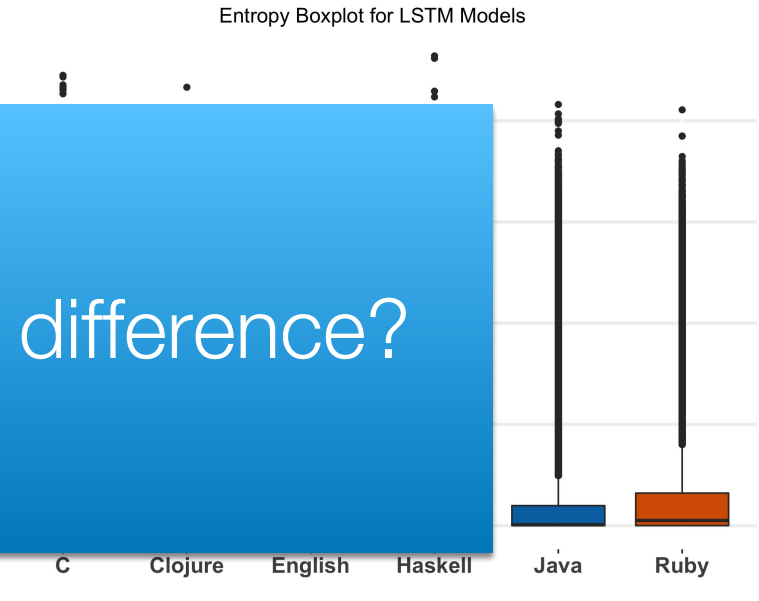


# Code is very repetitive and predictable compared to English

**And this can be found regardless of the language model used…**




**Or regardless of the type of programming language selected…**




What causes the difference?

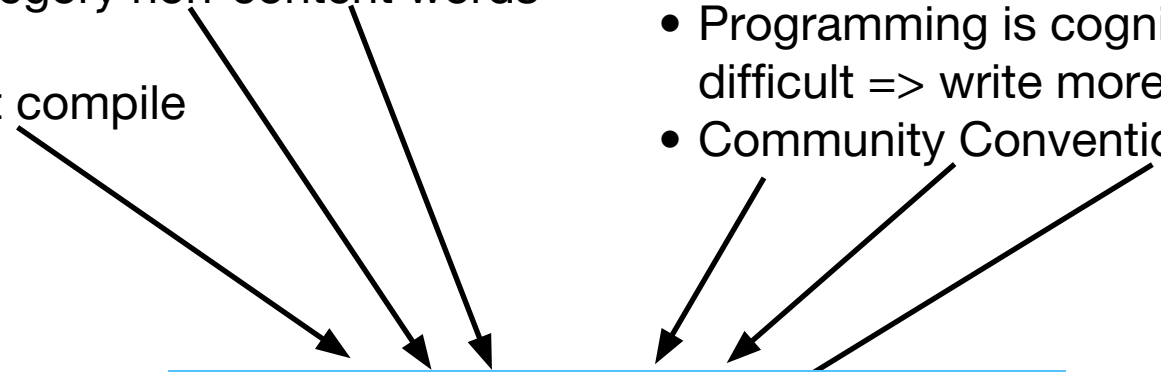
# Two Categories of Theories

## Syntactic/Structural

- Code grammar is unambiguous
- Closed category non-content words usage
- Code must compile

## Social/Cognitive

- Different requirements of technical tasks
- Programming is cognitively more difficult => write more repetitive
- Community Convention



Model?

# Two Categories of Theories

## Syntactic/Structural

- Code grammar is unambiguous
- Closed category non-content words usage
- Code must compile

## Social/Cognitive

- Different requirements of technical tasks
- Programming is cognitively more difficult => write more repetitive
- Community Convention

???

# Two Categories of Theories

## Syntactic/Structural

- Code grammar is unambiguous
- **Closed category non-content words usage**
- Code must compile

## Social/Cognitive

- Different requirements of technical tasks
- Programming is cognitively more difficult => write more repetitive
- Community Convention

# Open and Closed Vocabularies

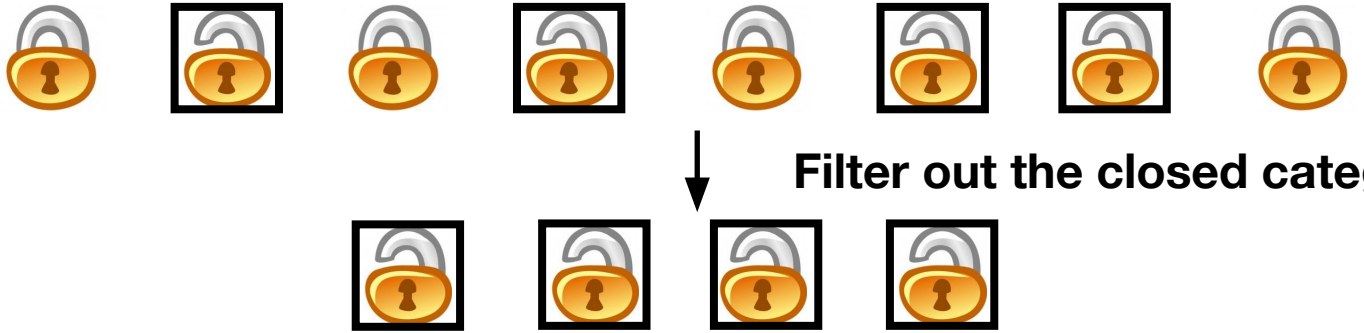
Open Category = Types of words to which new elements can be added freely.

- Java = variable names, types, method and class names (literals?)
- English = nouns, verbs, adjectives, etc…

Closed Category = Types of words to which new elements cannot be added.

- Java = operators, punctuation, etc.
- English = ‘stopwords’ (conjunctions, pronouns, articles, etc,), punctuation.

# A simple experiment




**Null Hypothesis:**
**If the differences between Code and English are due to**

1. **Code having many more closed category words…**
2. **And these words being more predictable…**

**Then we would find no difference in these reduced sequences.**

# Example Open/Closed Vocabulary texts

**Java**

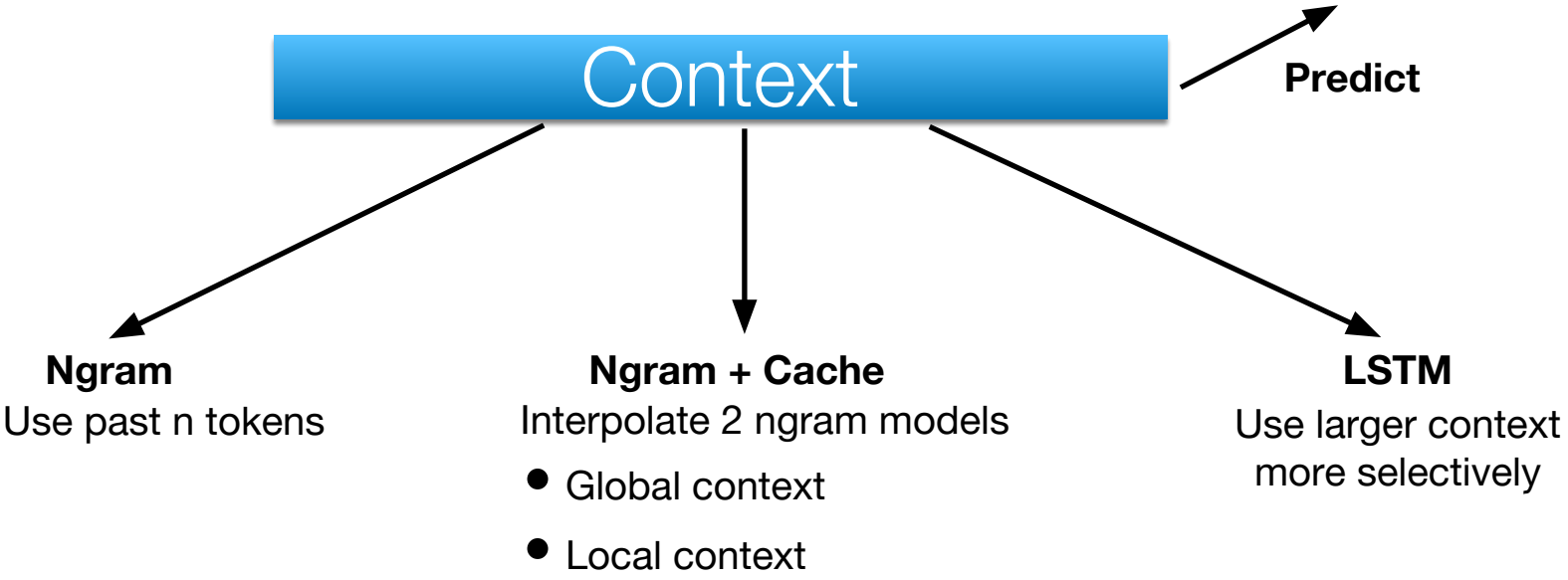
… String lines data split response setContentType response setCharacterEncoding int batchCount String s lines s s trim …

**English**

… Now 175 staging centers volunteers coordinating get vote efforts said Obama Georgia spokeswoman Caroline Adelman …

# Language Models

… headers . add ( name , new AsciiString ( tmp ) ) ; …



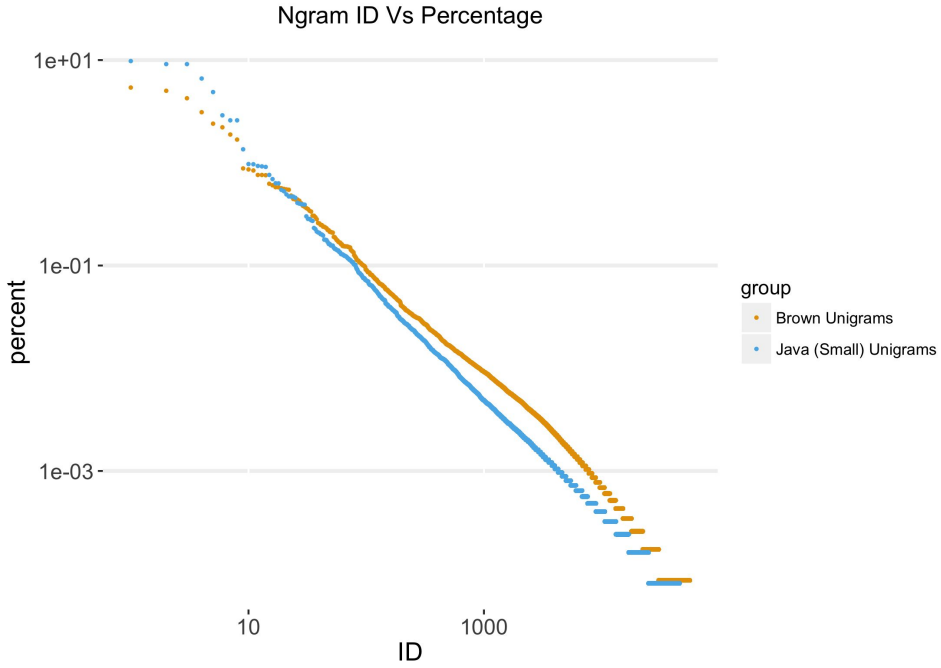

Entropy: Measure of Predictability. How many bits are necessary to represent the information? How surprising is the next token to the model?

# Zipf Plots

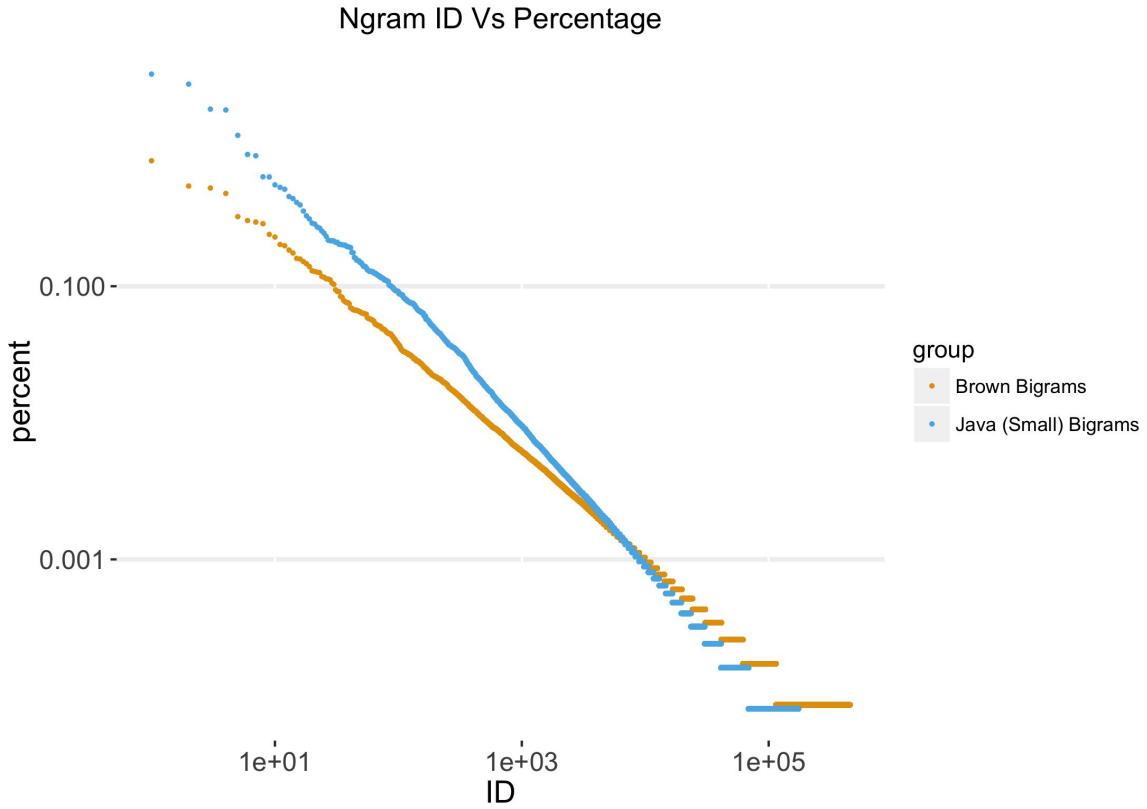
A commonly used plot for examining vocabulary distribution.

- Order all tokens in decreasing order
- Compare this rank (x-axis) against the tokens frequency.

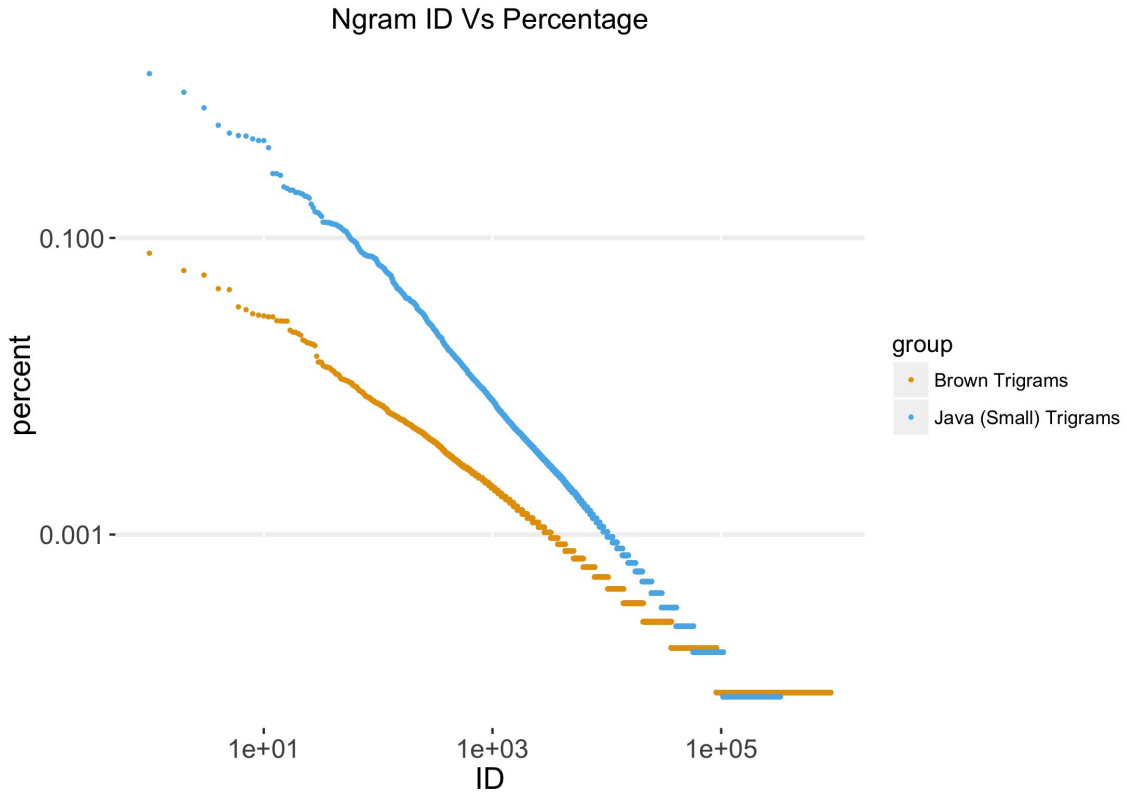
**Idea:** Extend the notion of this plot from single tokens to *sequences* of tokens => provides another way to measure language repetition.



# Zipf Plots (Bigrams)



# Zipf Plots (Trigrams)

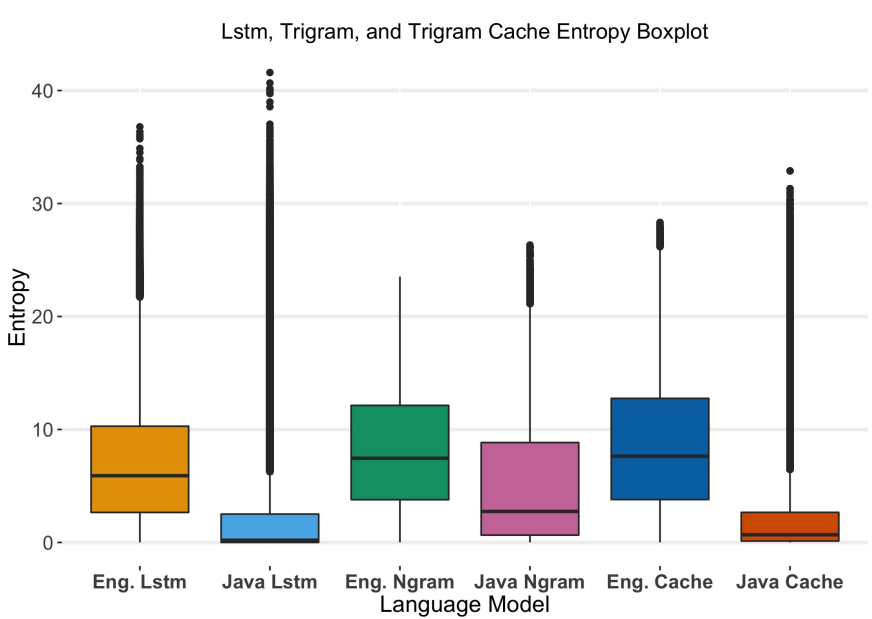


# Corpora Used

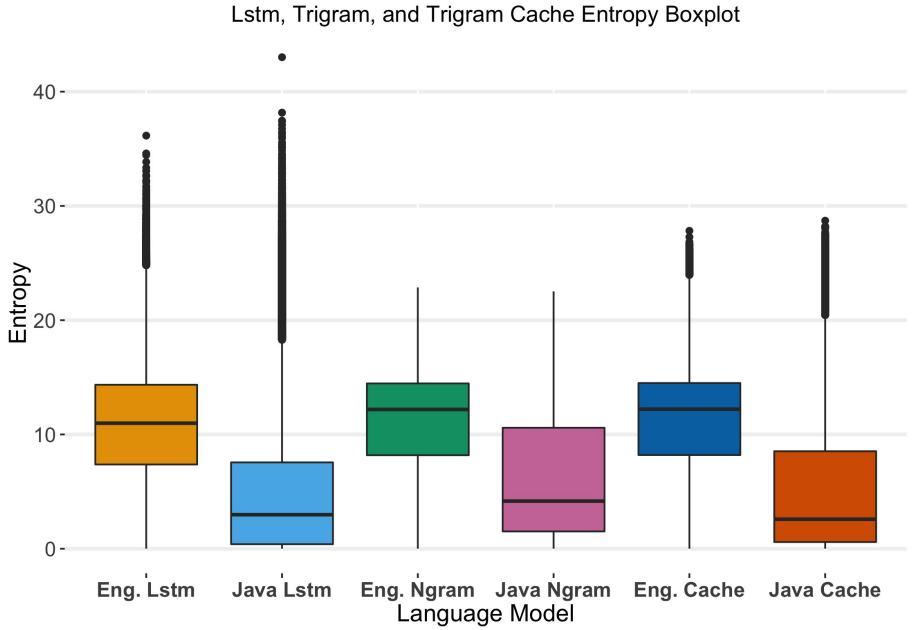
| | Tokens | Open Category (Ignoring Literals) | Open Category (With Literals) |
|---|---|---|---|
| English | 15708917 | 8340284 (53.1%) | — |
| Java | 16797357 | 5959414 (35.5%) | 6469474 (38.5%) |
| Haskell | 19113708 | 8569986 (44.8%) | 10803544 (56.5%) |
| Ruby | 17187917 | 3837434 (22.3%) | 8992955 (52.3%) |
| Clojure | 12553943 | 3283260 (26.2%) | 6286549 (50.1%) |
| C | 14172588 | 3707085 (26.2%) | 5846097 (41.2%) |

# Results: Language Models

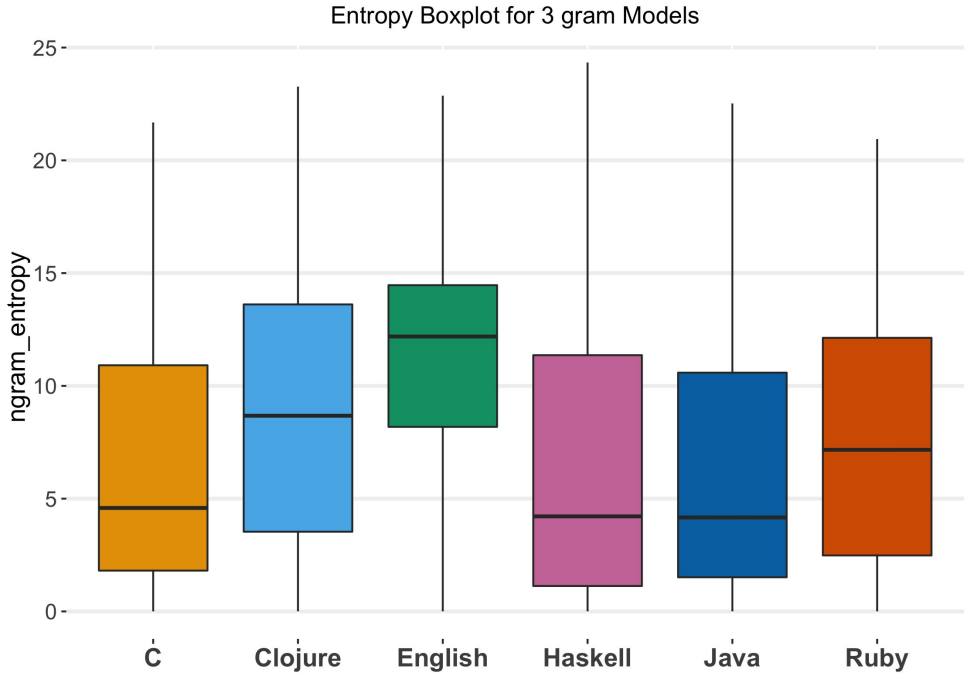
## All Tokens

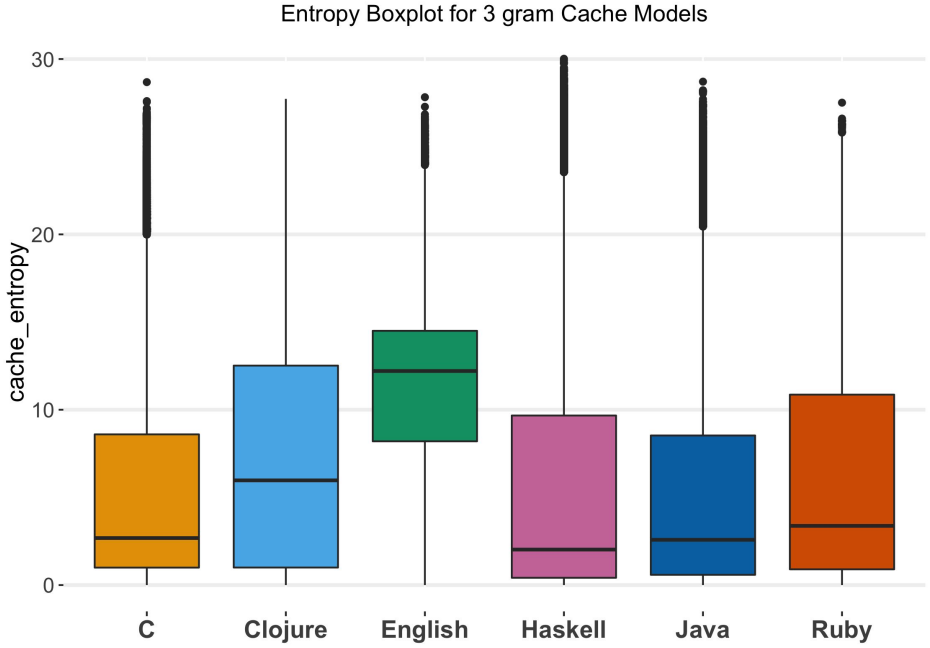



## Open Category Tokens

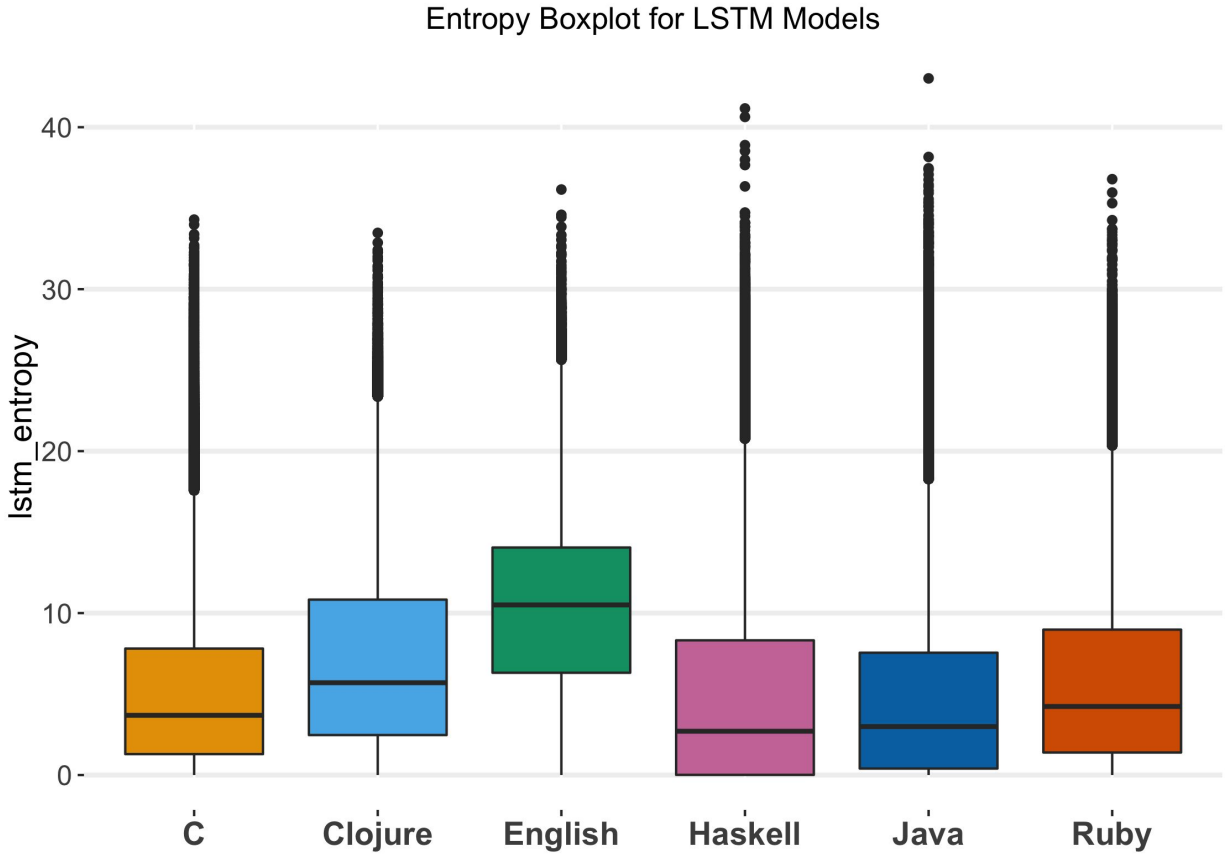


# Results: Language Models

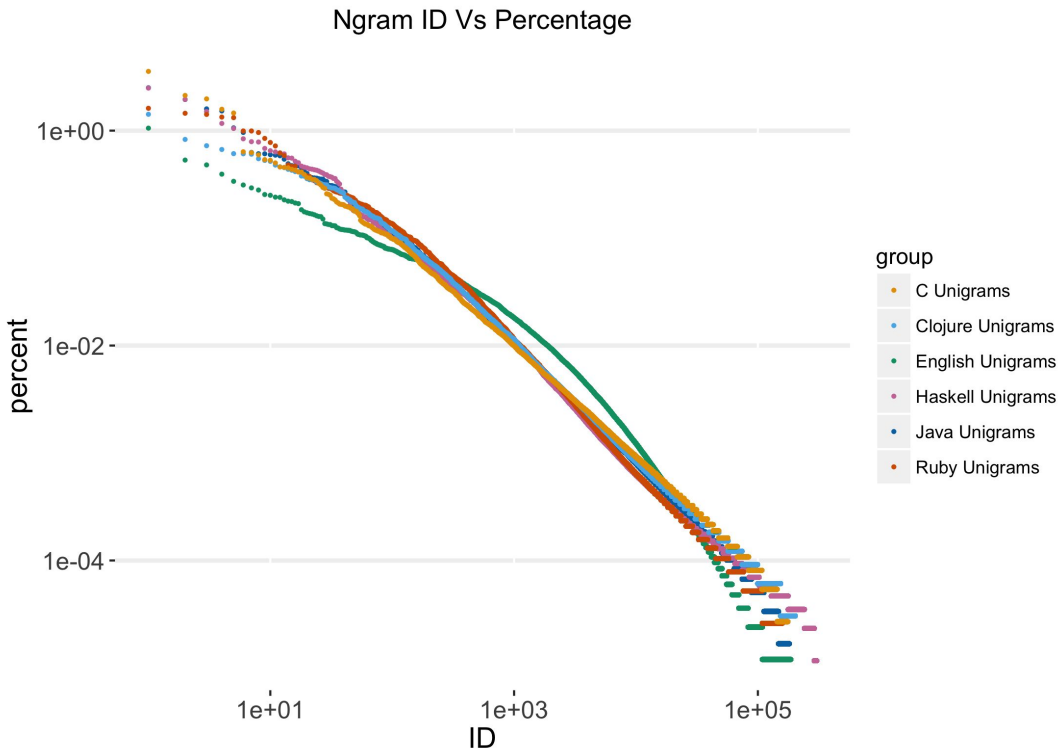


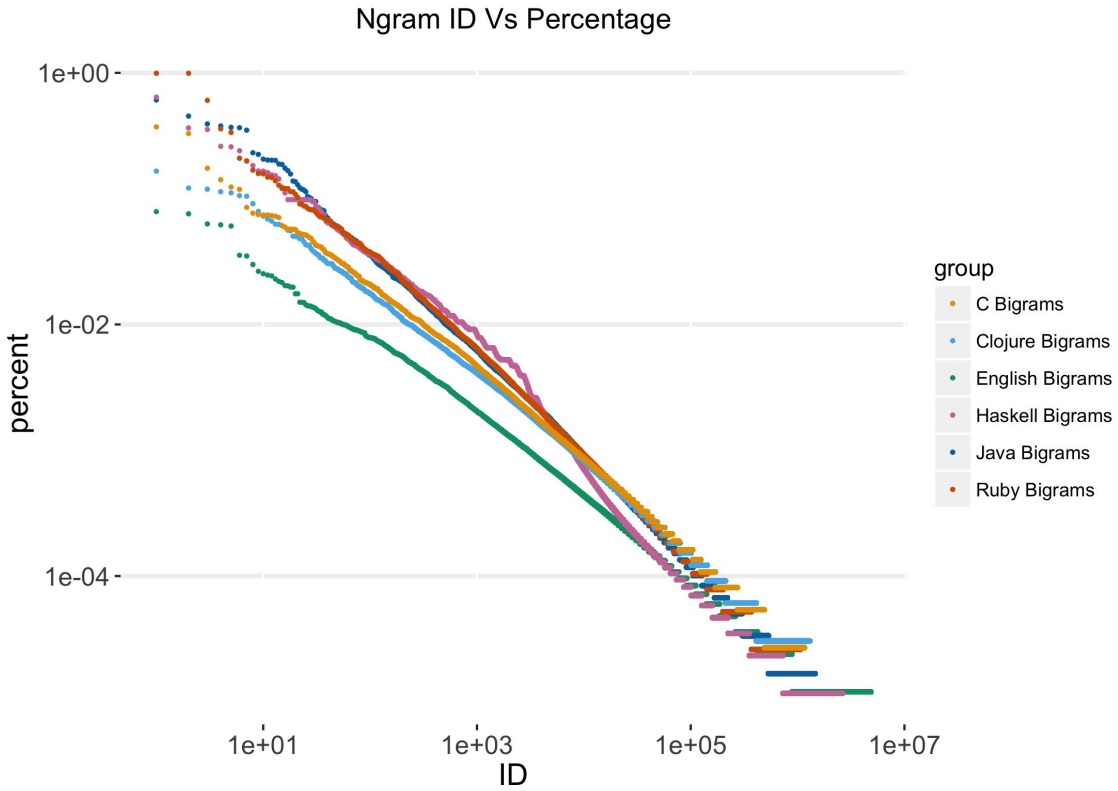
# Results: Language Models



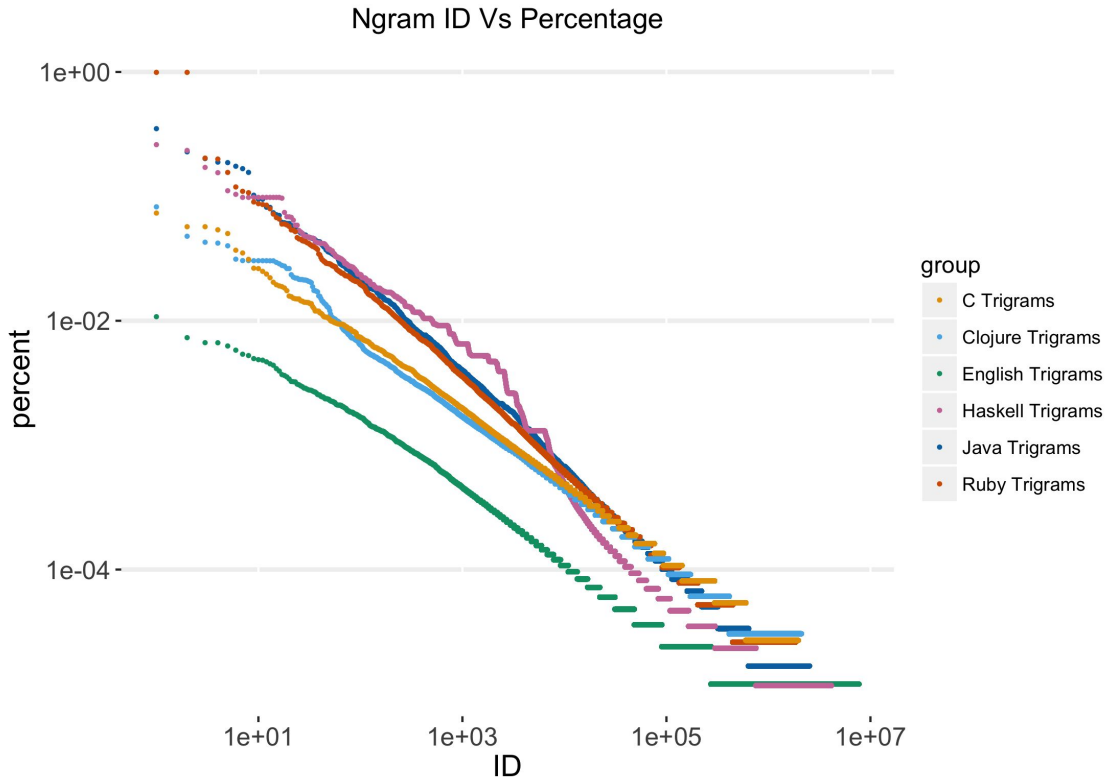
# Results: Zipf Plots (Unigrams)



# Results: Zipf Plots (Bigrams)



# Results: Zipf Plots (Trigrams)



# Frequent Trigram Examples

| | | |
|---|---|---|
| 0 | President Barack Obama | 899 |
| 1 | New York City | 610 |
| 2 | New York Times | 556 |
| 3 | George W Bush | 555 |
| 4 | * * * | 523 |
| 5 | Quote Profile Research | 484 |
| 6 | forward looking statements | 449 |
| 7 | 5 per cent | 438 |
| 8 | NEW YORK AP | 414 |
| 9 | guardian co uk | 406 |
| 10 | published guardian co | 402 |
| 11 | year old man | 390 |
| 12 | NEW YORK Reuters | 376 |
| 13 | two years ago | 366 |
| 14 | President George W | 344 |
| 15 | first time since | 329 |

| | | |
|---|---|---|
| 0 | com google common | 13528 |
| 1 | google common collect | 8003 |
| 2 | org apache cassandra | 7717 |
| 3 | com badlogic gdx | 7286 |
| 4 | int i i | 5665 |
| 5 | org eclipse debug | 5587 |
| 6 | Exception com google | 3341 |
| 7 | testCase com google | 3335 |
| 8 | org elasticsearch common | 3252 |
| 9 | eclipse debug internal | 2589 |
| 10 | java util concurrent | 2526 |
| 11 | org gradle api | 2357 |
| 12 | debug internal ui | 2351 |
| 13 | org apache thrift | 2343 |
| 14 | org nd4j linalg | 2136 |
| 15 | io netty handler | 2063 |

# Results Discussion

- The differences in repetition observed between English and programming languages are not merely due to the presence closed category syntactic structural words.

- In fact, the difference between them almost always *increases* when looking at only the open category words.

# Other Experiments

- Comparing Parse Trees
  - The effect of ambiguous vs. unambiguous grammars
  - Findings: the restrictive grammar explains some, but not all of the differences.
- Comparison to Technical and English Learner Texts
  - Both display trends away from generic English and more like Code.