Intelligent Code Editor

sdmay20-46 (Spring 2020) — Client and Advisor: Dr. Ali Jannesari

Keaton Johnson, Jonathan Novak, Matthew Orth, Garet Phelps, Isaac Spanier, John Jago

Motivation

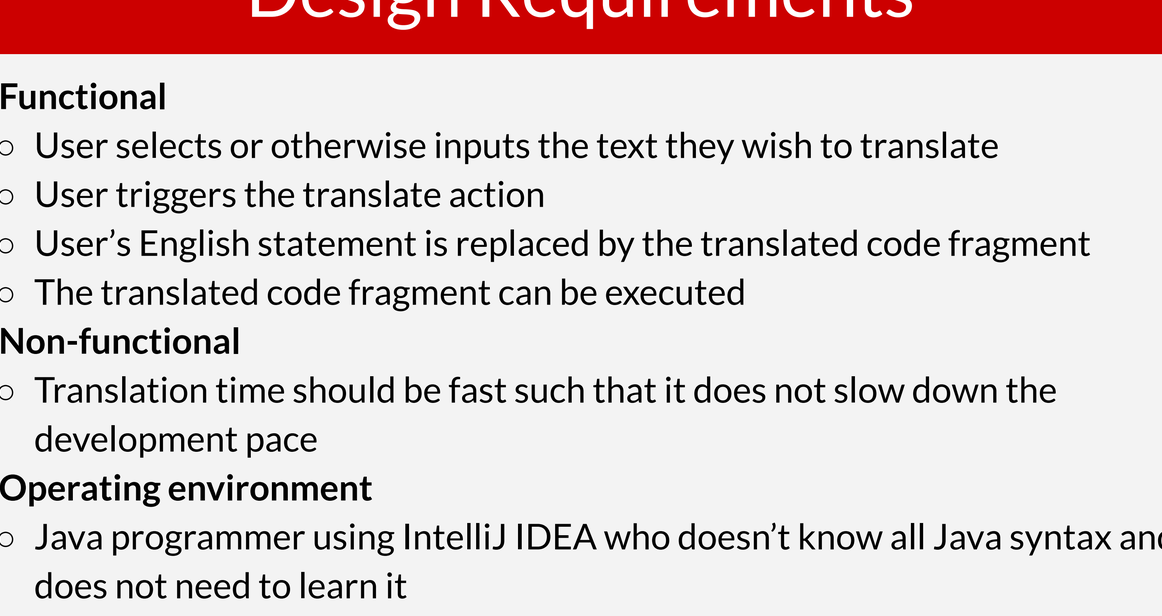
- **Problem:** Programming is now used in fields like statistics, but people in those fields don't need to know exact syntax of programming languages
- **Solution:** Neural machine translation model trained on Java method invocations (and an IDE plugin for the user)
- **Intended uses:** Translate English descriptions of Java method invocations
- **Engineering Standards and Design Practices:** IEEE 1028-2008, IEEE 16326-2009, IEEE 1008-1987, Agile Workflow, Test Driven Development

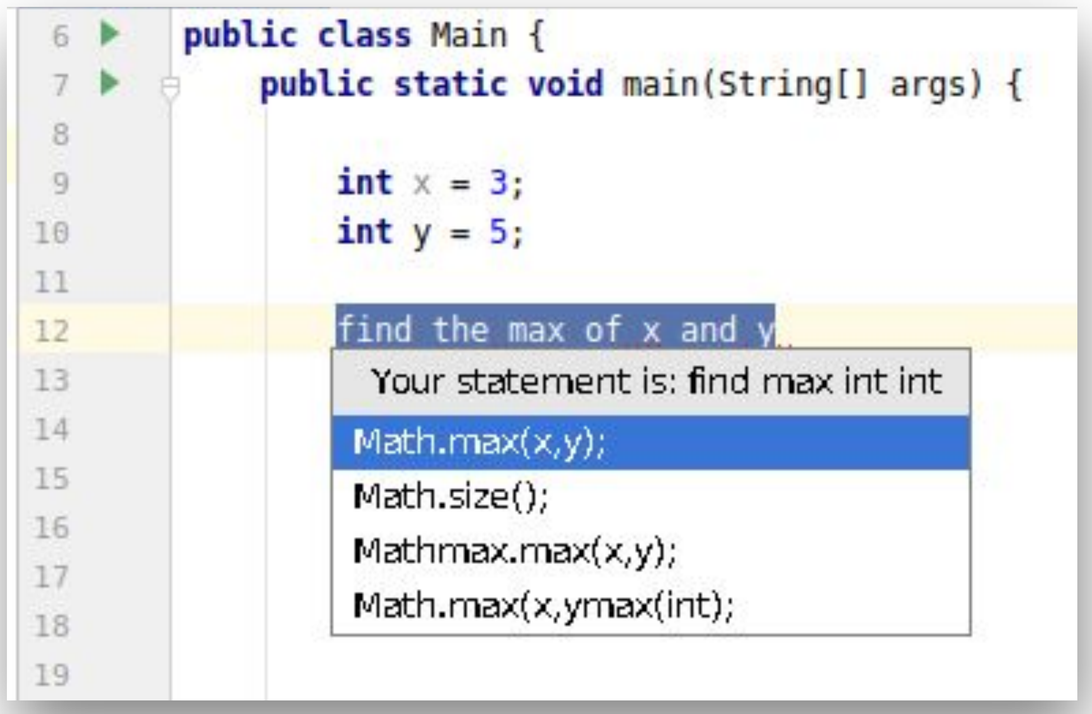
Design Requirements

- **Functional**
  - User selects or otherwise inputs the text they wish to translate
  - User triggers the translate action
  - User's English statement is replaced by the translated code fragment
  - The translated code fragment can be executed
- **Non-functional**
  - Translation time should be fast such that it does not slow down the development pace
- **Operating environment**
  - Java programmer using IntelliJ IDEA who doesn't know all Java syntax and does not need to learn it

Design Approach

- **User Interface**
  - Upon triggering translation, convert method parameters to their types
  - Preprocess raw English and send to translation engine
  - Displays the top expected Java code translation, most probable to least
- **Preprocessing**
  - Use NLTK to convert original statement into "verb-noun" format, present tense, and lowercase
- **Dataset**
  - One file for the natural language statements (in "verb-noun" format with parameters converted to types)
  - One file for corresponding Java code translations
- **Classification engine**
  - Train a model using the dataset
  - Transformer model architecture
- **Translation engine**
  - Run preprocessed user input through trained model
  - Returns top expected code translations to the user interface





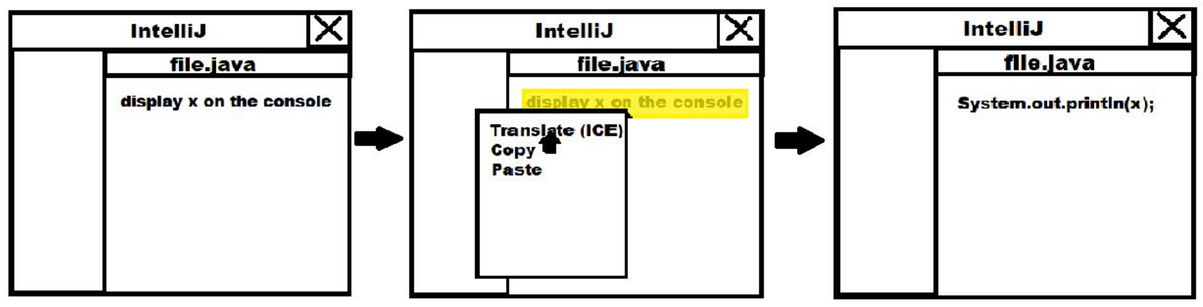
Technical Details

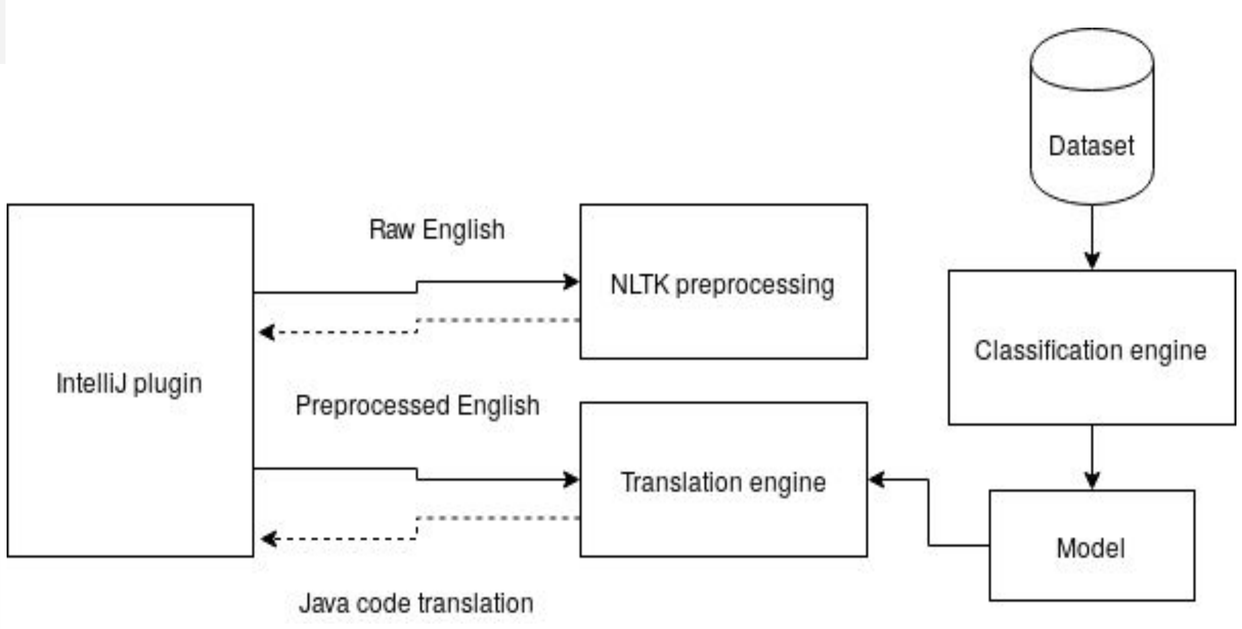
- **IntelliJ Platform SDK** for creating the plugin
- **Natural Language Toolkit (NLTK)** for preprocessing the input natural language statement into "verb-noun" format
- **OpenNMT-py** for implementing the translation from natural language to Java code
- AWS S3 Bucket, EC2 and Lambda servers for hosting translations
- **Programming languages**
  - Java: IntelliJ plugin, dataset preprocessing for mined Java code
  - Python: Preprocessing and classification/translation engine
  - C#: Dataset mining with Octokit

Testing

Unit: IntelliJ plugin (JUnit 5)
System: End-to-end translation
Acceptance: User experience

Translation Results
Accuracy: 50-60%
BLEU: 30-40





Acknowledgements We would like to thank Professor Ali Jannesari for his guidance on this project not only as the primary client but also as the faculty advisor. We would also like to thank Jannesari's PhD student, Hung Phan, for sharing what he has learned from his research to guide our project.