Comparing Natural Language Embeddings for Libc Functions as Rich Labels

Bachelor defense

Ruben Triwari

Ludwig Maximilian University Munich

19, February 2025

Outline

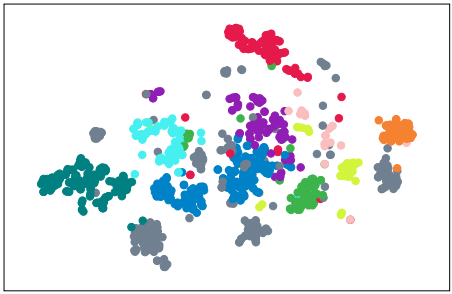
Motivation & Research Objective

Methodology

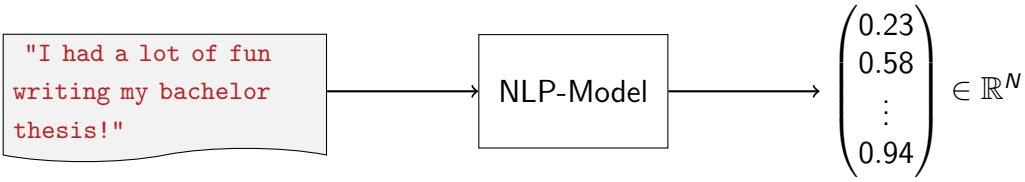
Results

Limitations

Conclusion & Future Work

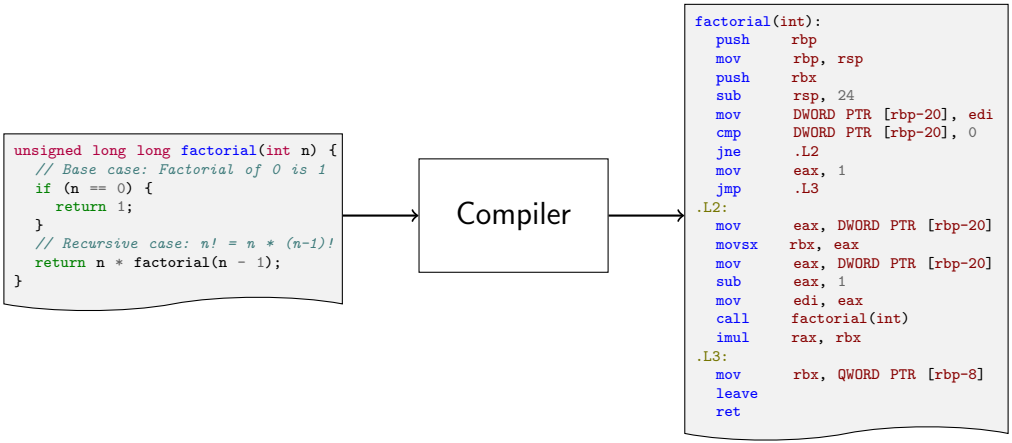


Motivation



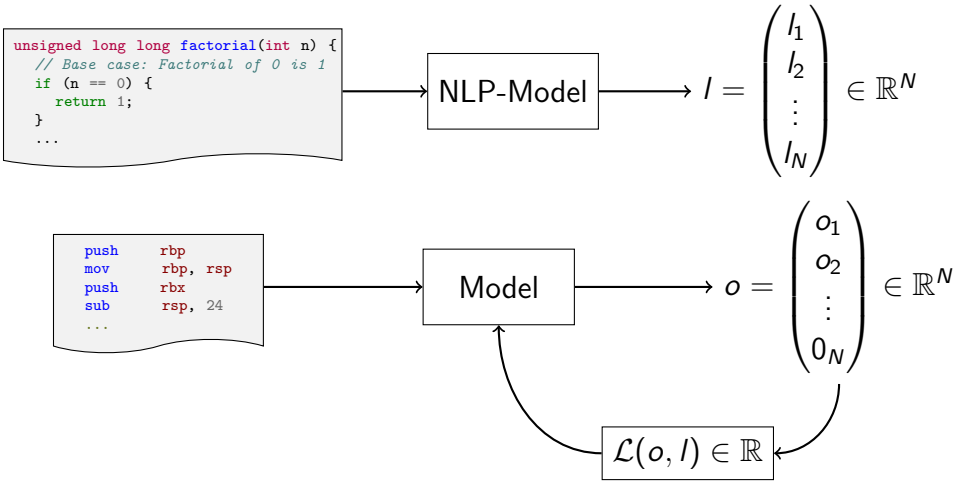
- $\rightsquigarrow$ Encoding natural language was a huge factor in recent nlp advancements
- $\rightsquigarrow$ Information described as a vector can be used in many downstream task
- $\rightsquigarrow$ That motivates encoding binary code and describing them as a vector
- $\rightsquigarrow$ That motivates using NLP tools to encode binary code

Motivation



$\rightsquigarrow$ Compiler removes important inforamtion in natural language

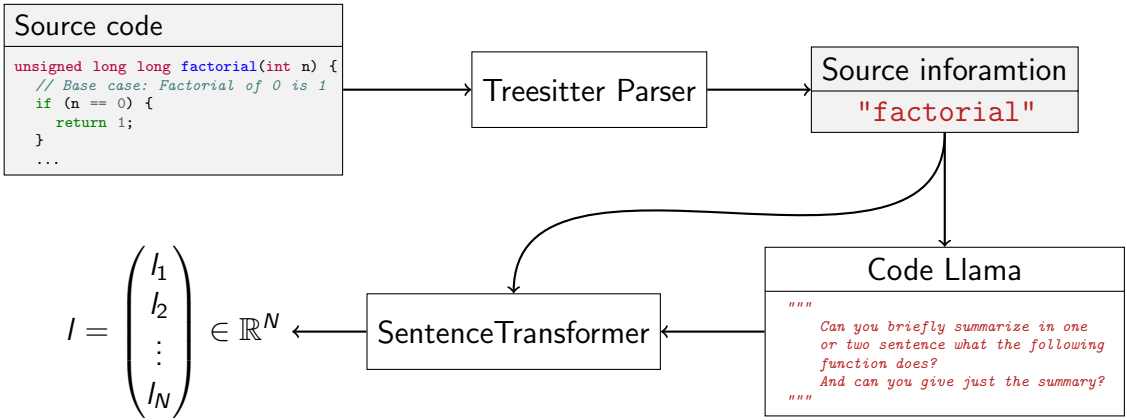
Motivation


unsigned long long factorial(int n) {
// Base case: Factorial of 0 is 1
if (n == 0) {
return 1;
}
...
NLP-Model
$I = \begin{pmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{pmatrix} \in \mathbb{R}^N$
push rbp
mov rbp, rsp
push rbx
sub rsp, 24
...
Model
$o = \begin{pmatrix} o_1 \\ o_2 \\ \vdots \\ 0_N \end{pmatrix} \in \mathbb{R}^N$
$\mathcal{L}(o, I) \in \mathbb{R}$

Research Objectives

- Compare diffrent approaches generating an Embedding with NLP tools
  1. Embed function names with SentenceTransformer
  2. Embed function comments with SentenceTransformer
  3. Embed Code-Llama code summaries with SentenceTransformer
- Compare NLP approach to the existig Code2Vec Model
- Propose a new way comparing embedding spaces

Architecture



Expert Survey

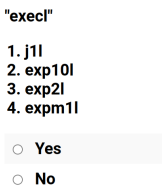


Figure: Positve exmaple

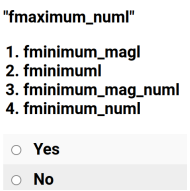


Figure: Negative exmaple

Ergebnisse der Expertenbefragung				
Strategie	Code-Llama-Erklärungen	Funktionsnamen	Funktionskommentare	*Code2Vec*
Score	0.596	0.532	0.433	0.321

Embeddings space comparison

$$\texttt{compare}(u, v)_k = \frac{1}{G_k} \sum_{i=1}^{k} \frac{\texttt{score}_k(u_i, i, v)}{log_2(i+1)} \in [0, 1]$$

where

$u, v \in \mathbb{N}^k$: Neighbor ranking of the same vector in diffrent spaces,

$$\texttt{score}_k(l, i, v) = \begin{cases} 1 & , \exists j \in \mathbb{N} : l = v_j \wedge i = j \\ \frac{1}{2} & , \exists j \in \mathbb{N} : l = v_j \wedge i \neq j \\ 0 & , \text{otherwise} \end{cases} , \; G_k := \sum_{i=1}^{k} \frac{1}{log_2(i+1)}.$$

Embeddings space comparison

$$\texttt{CMP}(A, B, k) = \frac{1}{N} \sum_{i=1}^{N} \texttt{compare}_k(NN_k(A_i, A), NN_k(B_i, B))$$

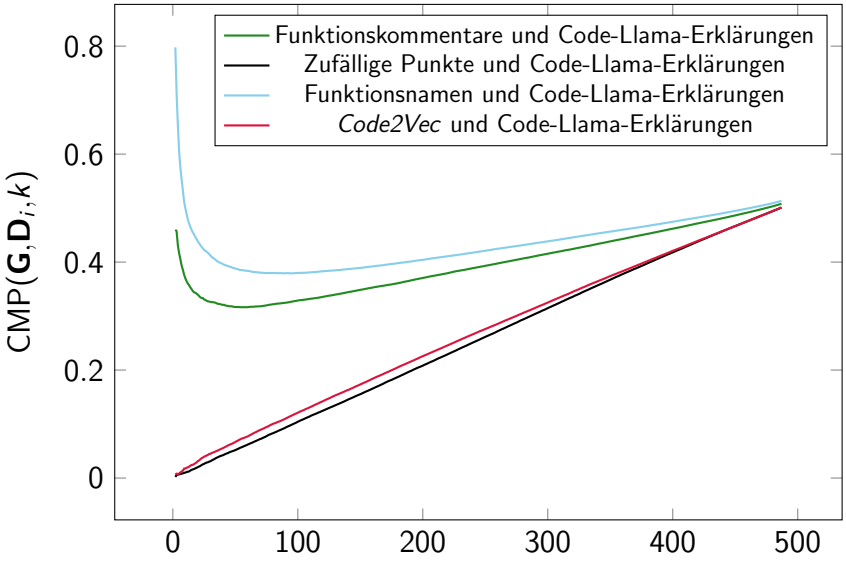
where

$A, B \in \mathbb{R}^{N \times l}$: Embedding space with N vectors of length$l$

$\texttt{NN}_k(A_i, A)$: k nearest neighbors from vector with index i in A

$k \in \mathbb{N}$: Amount of vectors we include in one neighborhood relation

Embeddings space comparison



Evaluation with T-SNE

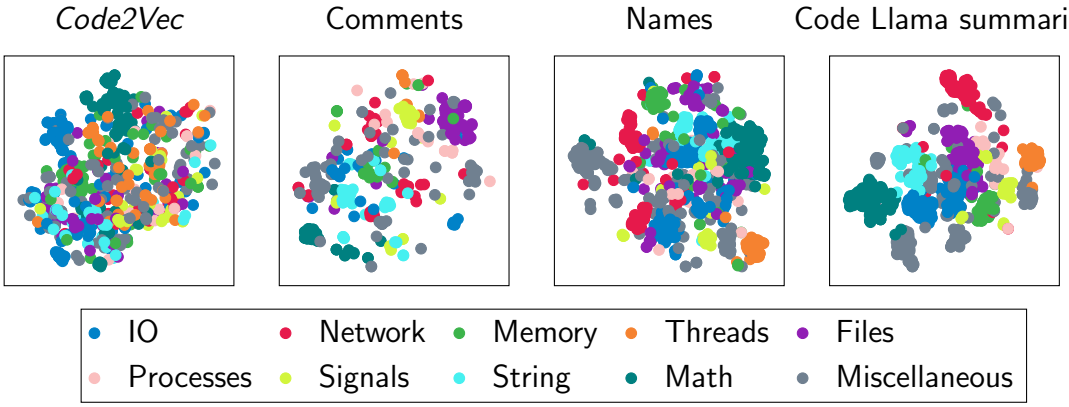


Figure: Depicted are the *t-SNE* output vectors with perplexity $P = 30$.

Function names

Abbreviations can potentially confuse the SentenceTransformer:
Example function `lchmod`:

$$\texttt{l} \leftrightarrow \text{link}, \quad \texttt{ch} \leftrightarrow \text{change}, \quad \texttt{mod} \leftrightarrow \text{file mode}.$$

Nearest neigbors in function space:

$$\texttt{lchmod} \leftrightarrow (\texttt{lcong48}, \ \texttt{fchmodat}, \ \texttt{coshl}, \ \texttt{cacoshl})$$

$\rightsquigarrow$ In categories:

$$\texttt{files} \leftrightarrow (\texttt{math}, \ \texttt{files}, \ \texttt{math}, \ \texttt{math})$$

function names

Example function `lchmod`:

$$\text{l} \leftrightarrow \text{link}, \quad \text{ch} \leftrightarrow \text{change}, \quad \text{mod} \leftrightarrow \text{file mode}.$$

Nearest neigbors in code llama summary space:

$$\texttt{lchmod}$$

$$\leftrightarrow \big(\texttt{fchmodat},\ \texttt{fchownat},\ \texttt{euidaccess},\ \texttt{\_\_file\_change\_detection\_for\_stat}\big)$$

$\rightsquigarrow$ In categories:

$$\texttt{files} \leftrightarrow \big(\texttt{files},\ \texttt{files},\ \texttt{files},\ \texttt{files}\big)$$

function comments

Comments are not always directly about the code:
Example functions `rand` and `rand_r`:

rand $\leftrightarrow$ Return a random integer between 0 and RAND_MAX.
rand_r $\leftrightarrow$ This algorithm is mentioned in the ISO C standard, here extended for 32 bits.

$\rightsquigarrow$ Cosine distance in comment and llama summary space

$$d_{\text{comment}}(\texttt{rand}, \texttt{rand\_r}) = 0.8544 \quad d_{\text{llama}}(\texttt{rand}, \texttt{rand\_r}) = 0.2216.$$

Code2Vec

- Also dpendend on the function names in the data set
- Bad results could be Explained by:
  1. Small data set
  2. C instead of Java
  3. Quality of names in the data set

Future Work

- Code Llama
  1. Is it necessary to use a large Model with 70B parameters?
  2. Can Large Language Models produce deterministic Ouput for this application?
  3. Is there a better Prompt?
- Comments
  1. Use inline Comments

Future Work

$$\texttt{CMP}(A, B, k) = \frac{1}{N} \sum_{i=1}^{N} \texttt{compare}_k(NN_k(A_i, A), NN_k(B_i, B))$$

$$\texttt{compare}(u, v)_k = \frac{1}{G_k} \sum_{i=1}^{k} \frac{\texttt{score}_k(u_i, i, v)}{log_2(i+1)} \in [0, 1]$$

- $\texttt{CMP}(A, B, k) \in [0, 1]$ function
  1. Is there an optimal value for k?
  2. Is there a better way to generate a neigborhood? (instead of K-Nearest-Neighbor)
  3. Is there a better way to aggregate the compare functions?

Conclusion

- Best strategies ranked:
  1. Code Llama summaries
  2. Function names
  3. Function comments
  4. Code2Vec
- Code Llama summary vectors for C source code downstream tasks
- Code Llama summary vectors can now be used to train a Model
- $\texttt{CMP}(A, B, k)$ function can be used to compare two embedding spaces from the same features Space

Discussion