LEARNING TO REPRESENT AND APPLY EDITS

Background
Edits are ubiquitous in both source code and language.

Key Idea
Language objective: maxmise the likelihood of generating the target \( x'_t \) given the input \( x' \) and the "ground-truth" edit \( \Delta(x'_t, x'_t) \)

Graph2Tree editors transduce the tree-structured input to the target tree via a sequence of tree-constructing actions (Yin et al., 2018)
- Source and target data (code) is represented as structured abstract syntax trees
- Source data is encoded using gated graph neural networks (Allamanis et al., 2018)

GITHUBEDITS CODE EDIT DATASET
- 110K pairs of C# code edits collected from GitHub commit histories
- Each edit involves at most three consecutive lines of code
- Shipped with parsed abstract syntax trees for each sample

Quality of Edit Representations
1. t-SNE Visualization
Systems trained on the noisy GitHubEdits, and tested on 2.8K edit pairs with known, labeled edit categories (16 in total). Examples:
- Before: \( x \rightarrow null \& \& x.StartsWith("a") \)
- After: \( x.StartsWith("a") \)

Graph-Tree Neural Editors
Motivation
The data to edit (e.g., source code) usually has strong underlying structure. How to utilize the structural information to better predict the edited output?

EXPERIMENTS

2. Edit Retrieval Task
Given a seed edit representation, retrieve and evaluate the relevance of retrieved neighboring edits

- Manually annotate the relevance with a scale of 3 (highly relevant, relevant, irrelevant)
- Compare with the upper-bound accuracy of using the oracle edit encoding \( \Delta(x'_t, x'_t) \)

Clustering Edits on GitHub Commits and Wikipedia Edit History
- Sampled edit clusters from GitHubEdits (omitting variable assignments for clarity)

EDIT REPRESENTATIONS
Sequential Encoding of Edits
- Use deterministic differencing algorithms to get alignments of tokens in the source and target, which are encoded using bi-LSTMs

Graph Encoding of Edits
- Source and target data is represented as syntax trees
- Add alignment edges between edited nodes. Encode the graph using graph neural networks (Li et al., 2016)