Towards Zero Knowledge Learning for Cross Language API Mapping

SMU SINGAPORE MANAGEMENT UNIVERSITY
School of Information Systems

Nghi D. Q. Bui

Motivation

- Language migration is important in software development
- API mapping is an indispensable step for the migration
- Existing works hinge on parallel data:
  - Require lots of human effort for data labeling
  - Require lots of parallel data for training

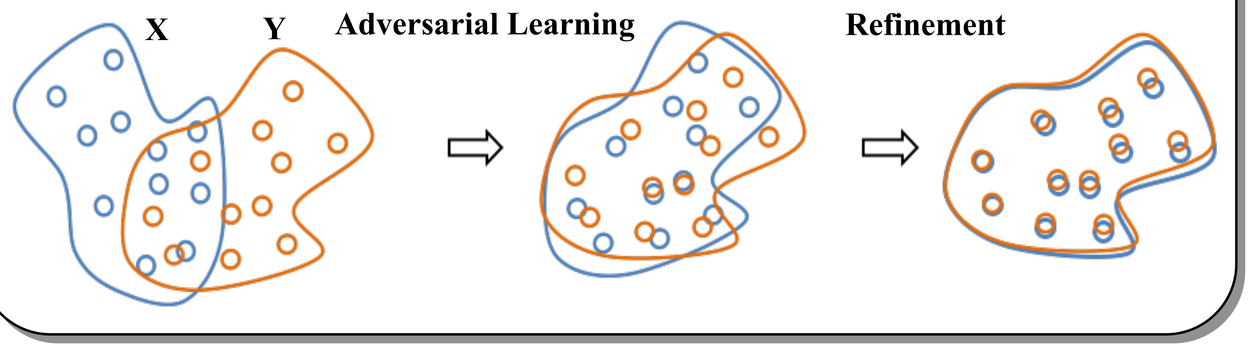
Our Vision:

*Effective API mapping **with less** reliance on parallel data (towards zero knowledge)*

Key Idea

Our Insight: Domain Adaptation

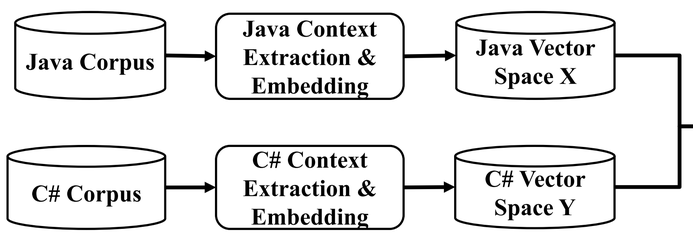
- Represent two languages as vector spaces X and Y
- Assume X and Y are similar in geometric arrangement
- Rotate source space X to align with target space Y



Approach

STEP 1: Code Embedding

- Extract API sequences with types
- Use Word2Vec for API embedding
- Construct two language vector spaces



STEP 2: Domain Adaptation

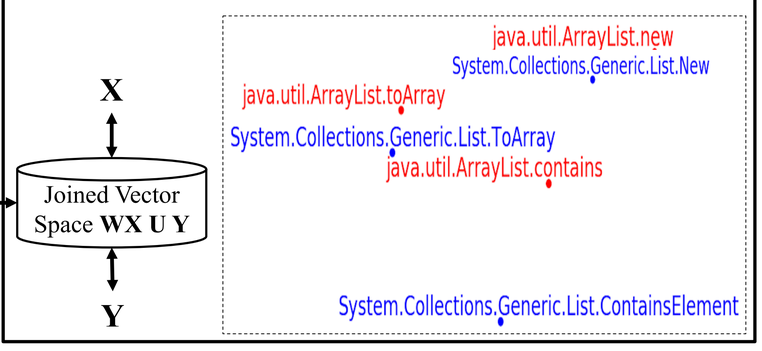
- Adapt the two vector spaces with a mapping matrix
- Learn a mapping matrix W, such that:

$$argmin_W ||WX - Y||$$

Adversarial Learning → Refinement

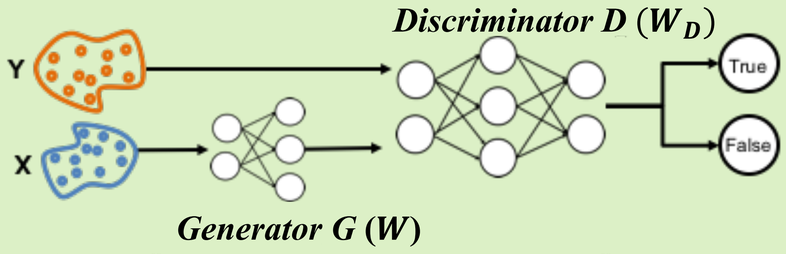
STEP 3: Query

- Query for mapping of an API x
- Mapping = nearest_neighbor(Wx)



Adversarial Learning

- Comprise of two neural networks:
  - ***Discriminator D*** (parameter W_D) maximizes likelihood of identifying an origin of an embedding
  - ***Generator G*** (parameter W) prevents the discriminator from doing so by making WX and Y as similar as possible

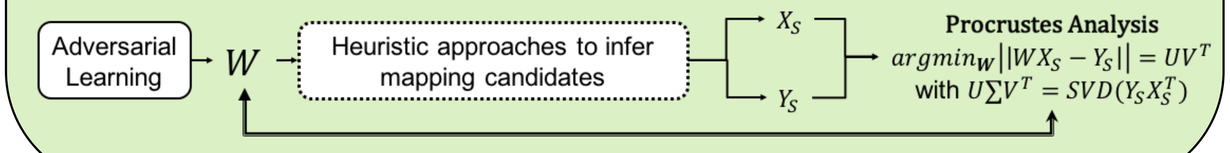


D: $L_D(W_D|W) = -\sum_{i=1}^{n} \log P_{W_D}(source = True|Wx_i) - \sum_{i=1}^{n} \log P_{W_D}(source = False|Wy_i)$

G: $L_G(W|W_D) = -\sum_{i=1}^{n} \log P_{W_D}(source = False|Wx_i) - \sum_{i=1}^{n} \log P_{W_D}(source = True|Wy_i)$

Refinement

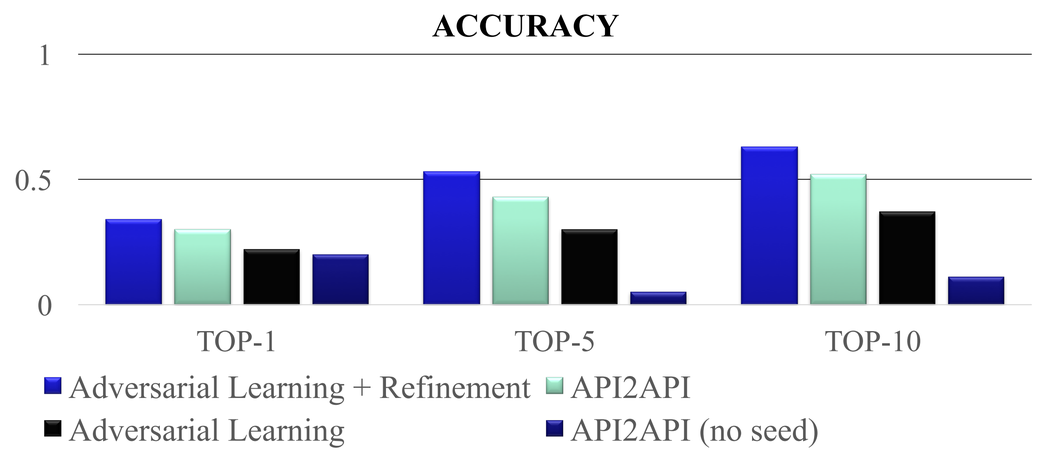
- Frequent APIs have higher mapping accuracy. They can be utilized as knowledge to refine Adversarial Learning
- Create mapping candidates X_S and Y_S based on following heuristics:
  - Signature-based mapping candidates
  - Top-K frequent mapping candidates
  - Cosine similarity threshold mapping candidates
- Derive W iteratively from Procrustes Analysis of X_S and Y_S



Training data does not need to be parallel and can be collected independently → Reducing human efforts

Evaluation

- **Data**: Java and C# repositories on Github, 2 million files
- **Baseline**: API2API [4]
- **Metric**: Top-k accuracy (k = 1,5,10)
- **Ground truth** : 860 mappings from Java2CSharp as the evaluation dataset, 2 folds validation



Future work

- Consider more advanced Adversarial Learning techniques
- Explore on API sequence mapping with zero knowledge
- Investigate on API mapping between languages that are different in geometric arrangement

References

[1] "*Statistical learning approach for mining API usage mappings for code migration*", by Anh Tuan Nguyen, Hoan Anh Nguyen, Tung Thanh Nguyen, and Tien N. Nguyen (ASE 2014).
[2] "*Mining API mapping for language migration*", by Hao Zhong, Suresh Thummalapenta, Tao Xie, Lu Zhang, and Qing Wang. (ICSE 2010)
[3] "*DeepAM: Migrate APIs with Multi-modal Sequence to Sequence Learning*", by Xiaodong Gu, Hongyu Zhang, Dongmei Zhang, and Sunghun Kim. 2017, (IJCAI 2017)
[4] "*API2API*: *Exploring API embedding for API usages and applications*", by Trong Duc Nguyen, Anh Tuan Nguyen, Hung Dang Phan, and Tien N. Nguyen. (ICSE 2017).