SI2 - ST1
Zincbase

Presented by:
João Santos nºmec 76912
João Ferreira nºmec 80305
Miguel Carvalho nºmec 80169

Introduction

Introduction

The knowledge representation tool chosen for the ST1 assignment was “Zincbase”.

Introduction

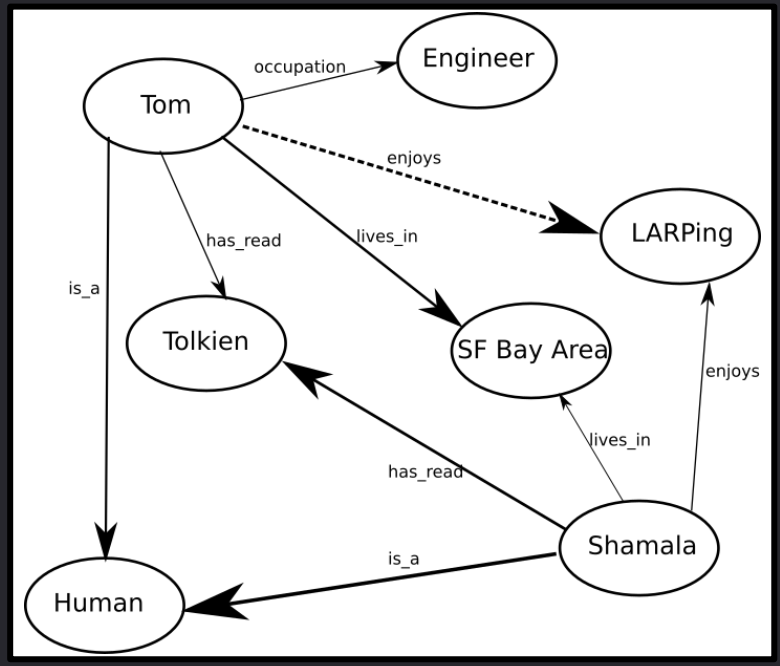
Zincbase is a state of the art knowledge base and simulation suite, combining neural networks with symbolic logic, graph search, and complexity theory.

Introduction

- Store and retrieve graph structured data efficiently.
- Provide ways to query the graph, including via graph neural networks.
- Simulate complex effects playing out across the graph and see how predictions change.
- It allows to compute the probability of a given query.

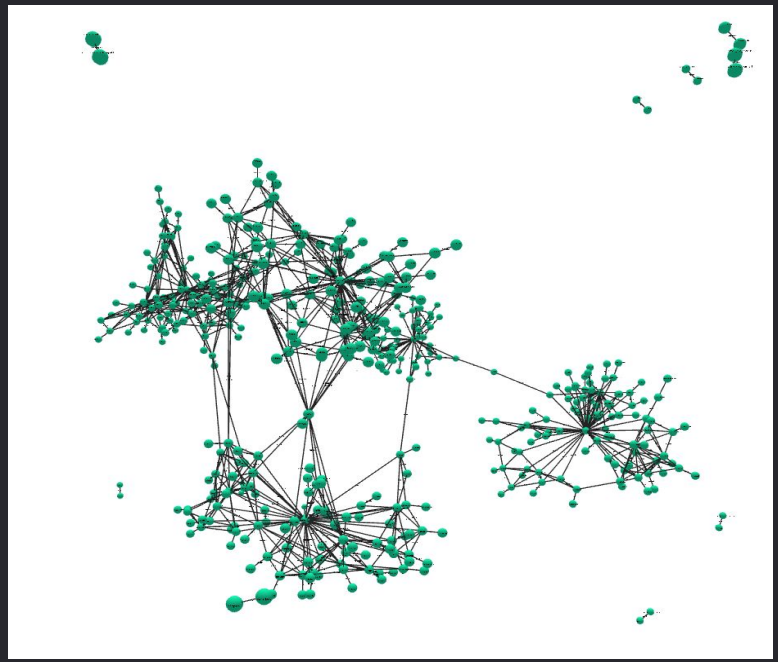
Introduction

It can be simple …



Introduction

… and it can be complex.



Prolog Integration

Facts - example fact-query

```
from zincbase import KB
kb = KB()
kb.store('eats(tom, rice)')
for ans in kb.query('eats(tom, Food)'):
    print(ans['Food'])
    # 'rice'
```

Facts - csv file

countries_s1_train.csv

portugal	locatedin	europe
portugal	neighbor	spain
portugal	bathedby	atlantic_ocean
lisboa	capitalof	portugal

Facts - store from file

```
from zincbase import KB
kb = KB()
kb.from_csv('countries_s1_train.csv', delimiter='\t')
for ans in kb.query('locatedin(Country, southern_europe)'):
    print(ans['Country'])
    # portugal, …
```

Facts - store from struct

```
from zincbase import KB

kb = KB()
eu_contries = ['austria', 'belgium', 'bulgaria', 'croatia',
'republic_of_cyprus', 'czech_republic', 'denmark', 'estonia', 'finland',
'france', 'germany', 'greece', 'hungary', 'ireland', 'italy', 'latvia',
'lithuania', 'luxembourg', 'malta', 'netherlands', 'poland', 'portugal',
'romania', 'slovakia', 'slovenia', 'spain', 'sweden']
for country in eu_contries:
     kb.store('partofEU('+ country +')')
```

Rules

```
# Capital of neighboring countries
kb.store('capitalofneighbor(Z, X) :- neighbor(X, Y), capitalof(Z, Y)')
# Capital of a neighbor that belongs to the european union
kb.store('capitalofneighborEU(Z, X) :- neighbor(X, Y), partofEU(Y), capitalof(Z, Y)')
# Country that belongs to a continent and is bathed by an ocean
kb.store('countriesofcontbathedby(Z, X, O) :- locatedin(Z, X), bathedby(Z, O)')
# Countries that are bathed by 2 specific oceans
kb.store('bathedby2(Z, O1, O2) :- bathedby(Z, O1), bathedby(Z, O2)')
```

Query

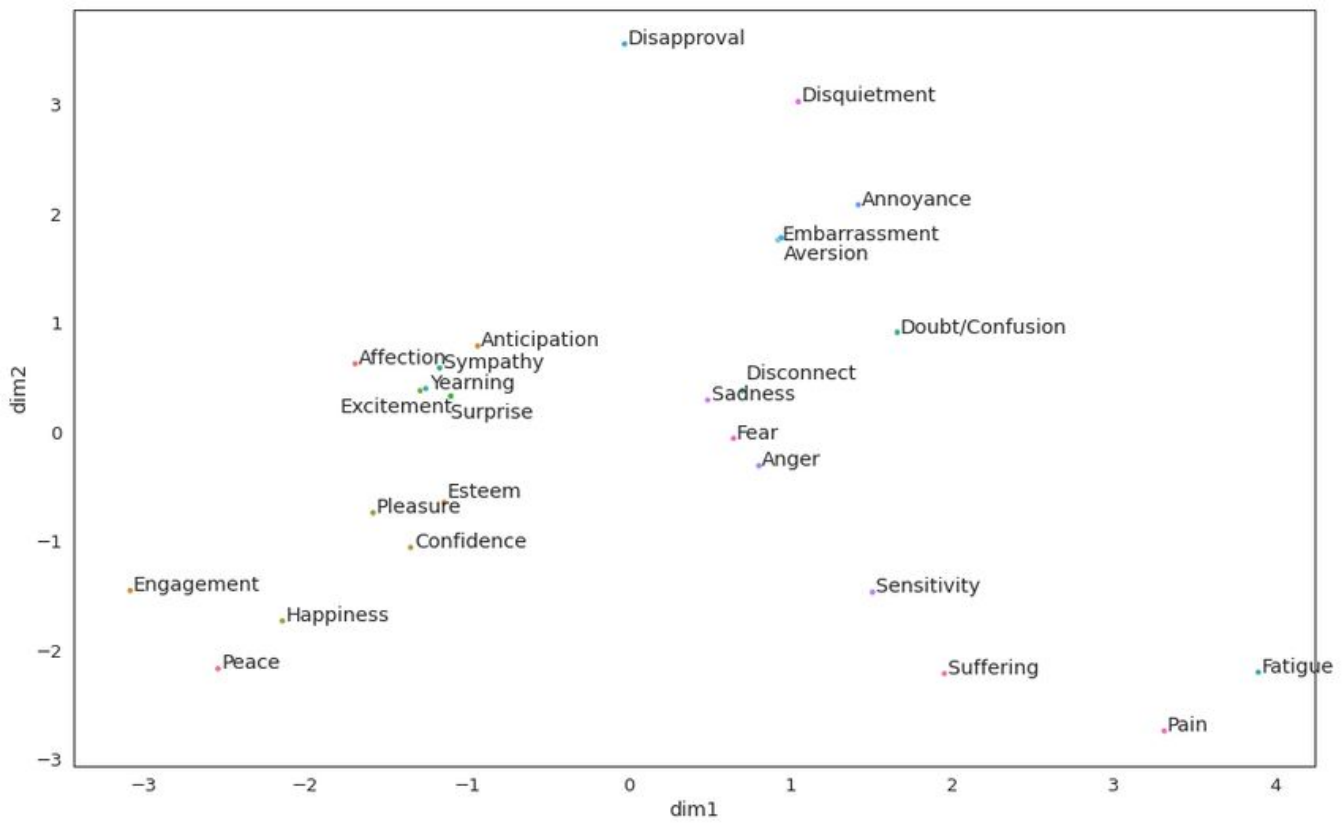
```
# Neighbor countries of the origin country of Madrid
for ans in kb.query('capitalofneighbor(madrid, Country)' ):
   print(ans)

# Capital of neighbor countries of Germany
for ans in kb.query('capitalofneighbor(City, germany)' ):
   print(ans)

# Country bathed by Atlantic and Pacific oceans
for ans in kb.query('bathedby2(Country, atlantic_ocean, pacific_ocean)'):
    print(ans)
```

Word Representation Learning

Representation



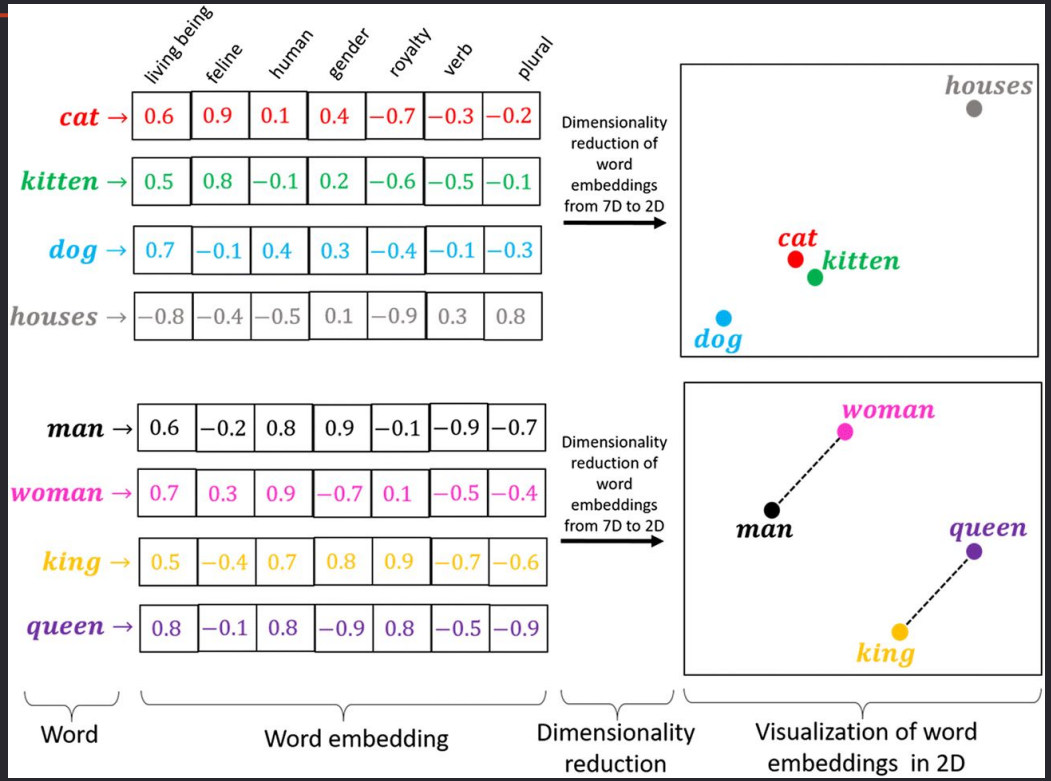
Learning

```
kb = KB()
kb.from_csv('./countries_train.csv', delimiter='\t')
kb.build_kg_model(cuda=False, embedding_size=40)
kb.train_kg_model(steps=4000, batch_size=1, verbose=True)
# get the entity which is more probable to be related to it based on the
relation
kb.get_most_likely('austria', 'neighbor', '?', k=2)
# estimates the probability of a given fact (triple)
kb.estimate_triple_prob('fiji', 'locatedin', 'melanesia')
# get the nearest entities in the vector space
kb.get_nearest_neighbors('united_kingdom', k=4)
```

Word embeddings

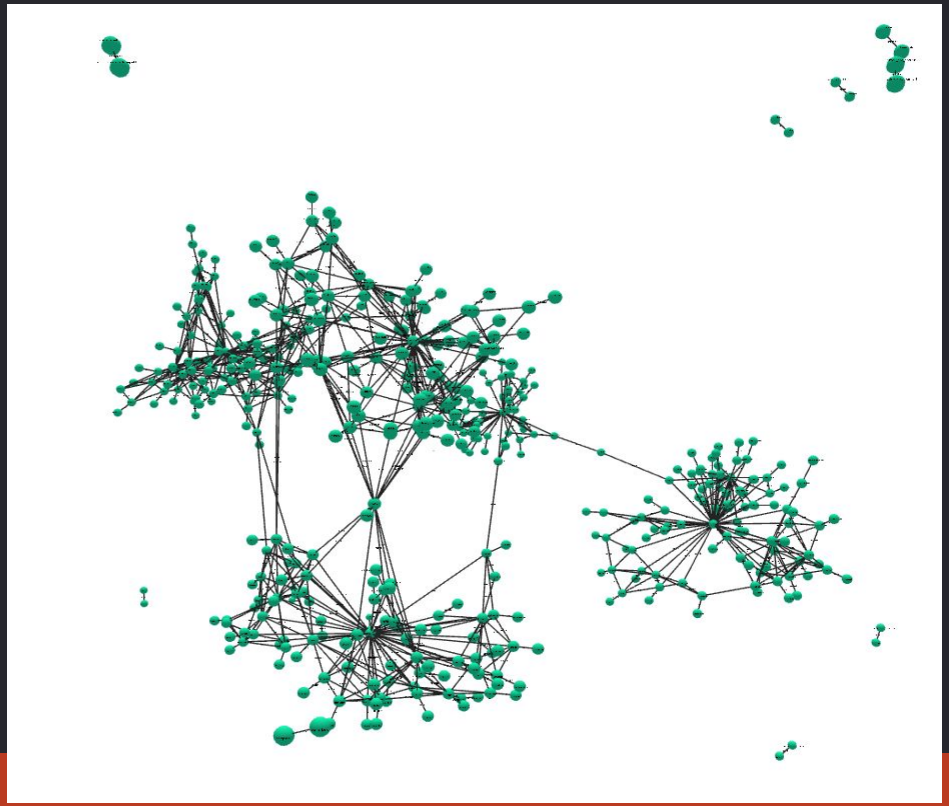
Zincbase generates word embeddings using neural networks from **pytorch** library.

Zincbase offers a simple and clean API for training.



Web UI

Web UI



Demo

Thank you!