Citation prediction challenge - Advanced Learning For Text and Graphs

~ A complete guide to alchemy ~

Project Presentation

22/02/2022

APAVOU Clément
BELKADA Younes
ZUCKER Arthur

MVA Master's students
École Normale Supérieure Paris-Saclay
firstname.lastname@ens-paris-saclay.fr

Introduction

Context

How to accurately predict a potential citation?

Challenges:

- We are given a very large graph (more than 1 million edges) of citation links between research papers
- How to extract meaningful information from that?
- How to not overfit?
- etc.

Methods

Raw ingredients: Dataset Creation and Split

For this challenge, the key is to find *the best dosage and combination of features*, aka the **ingredients**

Split by instances

- Consider all the positive pairs and randomly generate negative pairs
- We randomly split the generated instances using a trainval rate

Improvements

+ Make sure that the generated negative pairs are not linked together

+ Possible improvement: add data augmentation, but how?

Introduction

How we see this challenge

Are we doing alchemy?

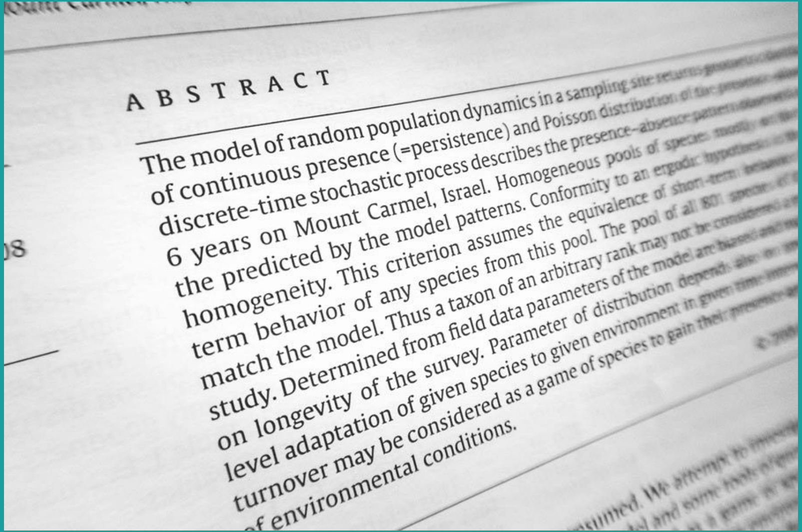
Challenges:

- There are several interesting approaches to extract features from a graph and texts
- The key is to find the best features and combination of them: we decided to become **alchemists**

NIPS 2017: *Machine Learning has become alchemy*

Methods

Ingredient 1: Abstracts!



ABSTRACT

The model of random population dynamics in a sampling site returns [illegible]
of continuous presence (=persistence) and Poisson distribution of the presence [illegible]
discrete-time stochastic process describes the presence–absence pattern [illegible]
6 years on Mount Carmel, Israel. Homogeneous pools of species mostly [illegible]
the predicted by the model patterns. Conformity to an ergodic hypothesis [illegible]
homogeneity. This criterion assumes the equivalence of short-term [illegible]
term behavior of any species from this pool. The pool of all 801 species [illegible]
match the model. Thus a taxon of an arbitrary rank may not be considered [illegible]
study. Determined from field data parameters of the model are biased and [illegible]
on longevity of the survey. Parameter of distribution depends also on [illegible]
level adaptation of given species to given environment in given time [illegible]
turnover may be considered as a game of species to gain their presence [illegible]
of environmental conditions.

Abstracts are arguably the richest feature in this problem

image source: https://writebetter.io/how-to-write-a-good-abstract/

Pre-trained Language models

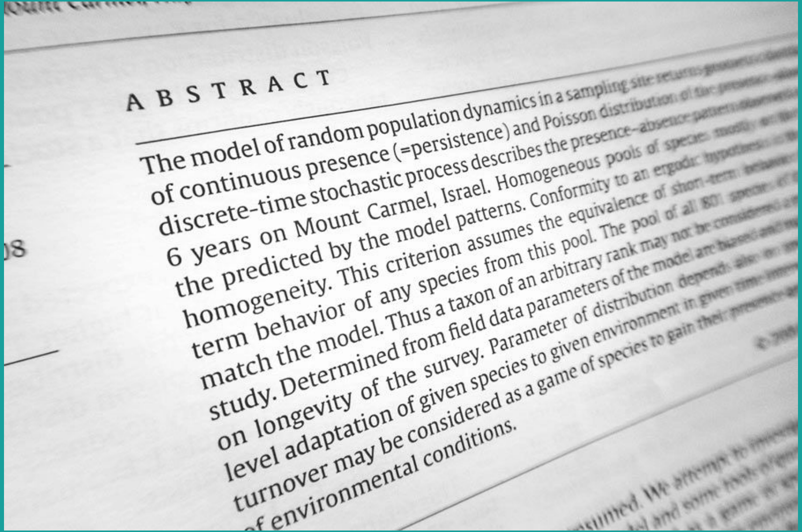
1- SciBERT: aka the BERT model trained on scientific articles

2- SPECTER: a LM trained on a scientific corpus for **link prediction**

Models available on: 🤗

Methods

Ingredient 1: Abstracts!



ABSTRACT

The model of random population dynamics in a sampling site
of continuous presence (=persistence) and Poisson distribution
discrete-time stochastic process describes the presence–absence
6 years on Mount Carmel, Israel. Homogeneous pools of species
the predicted by the model patterns. Conformity to an ergodic
homogeneity. This criterion assumes the equivalence of short-term
term behavior of any species from this pool. The pool of all
match the model. Thus a taxon of an arbitrary rank may not be considered
study. Determined from field data parameters of the model are biased
on longevity of the survey. Parameter of distribution depends
level adaptation of given species to given environment in given time
turnover may be considered as a game of species to gain their presence
of environmental conditions.

Abstracts are arguably the richest feature in this problem

image source: https://writebetter.io/how-to-write-a-good-abstract/

Pros

- Arguably the richest feature that can be used
- Several powerful pretrained language models available

Cons

- Slow to compute (needs an off-the-shelf script to precompute the features)
- Not aware of context (i.e, neighbors)

Methods

Ingredient 2: Handcrafted graph features

Extension from the provided baseline:

1. Degree of node 1
2. Degree of node 2
3. Sum of degrees
4. Absolute value difference of the degrees
5. Length of intersection of the neighbors (biased sorenson index)

→ Not hard to compute, these features are computed when initializing the dataset

An attempt to leverage the information from the graph

Methods

Ingredient 3: Contextual features using graphs and texts!

Can we merge information from graph and text?

Fuse graph and text!

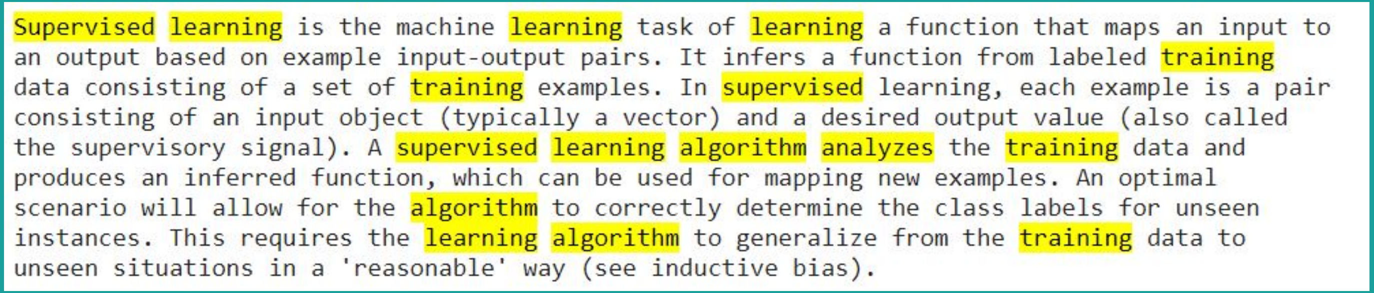
For each **node pair**:

1. Get the neighbor nodes (excluding the other one from the pair if it is cited)
2. Compute the SPECTER embeddings for each node
3. Get the mean representation of these nodes

This should give the model an accurate idea of the **context** of the neighbor nodes

Methods

Ingredient 4: Using keywords



Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It infers a function from labeled training data consisting of a set of training examples. In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a 'reasonable' way (see inductive bias).

The keywords should be an important indicator of the content of the scientific paper

Make use of KeyBERT to extract 10 keywords from each abstract

1. Get the 10 keywords for each abstract

2. For each pair of nodes, get the number of common keywords
   OR
3. Keywords mean embeddings from SPECTER

KeyBERT: https://github.com/MaartenGr/KeyBERT

Methods

Ingredients 5: Using Graph features from networkx.link_prediction

We make use of:

- jaccard coefficient
- adamic adar index
- soundarajan hopcroft index
- common neighbor centrality
- clustering coefficient
- sorenson index

Methods

Tools : How to find the best training strategy through ablation

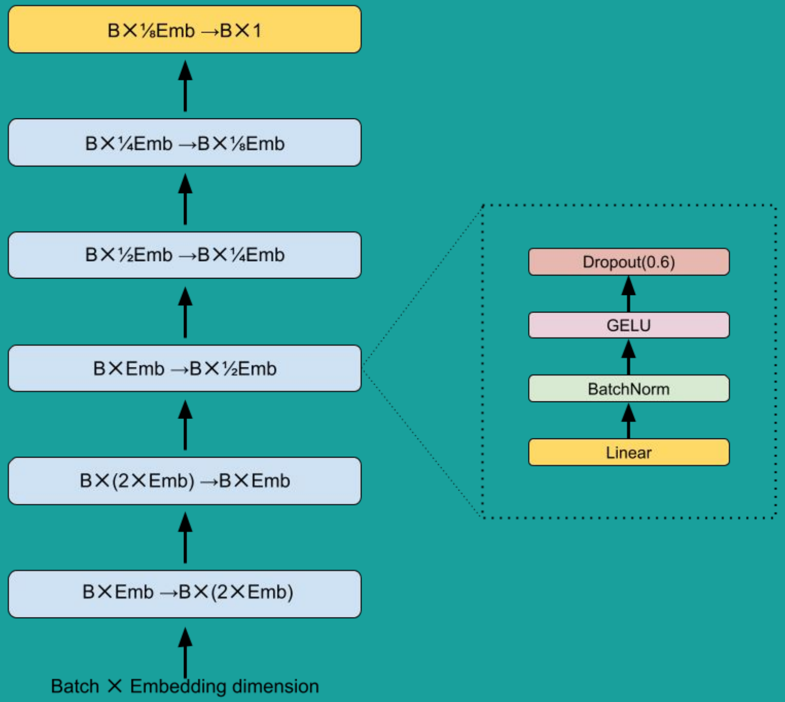
What are the best tools that we have to use for this challenge?

We tried a lot of architectures

- optimizers : Adam, Adamw, RMSProp, SGD
- activations : Relu, Elu, Gelu, Tanh
- norm layers : Batch, Layer
- Dropout : from 0 to 0.8
- Weight decay ...
- Small and big MLPs

Methods

Tools : Best tools

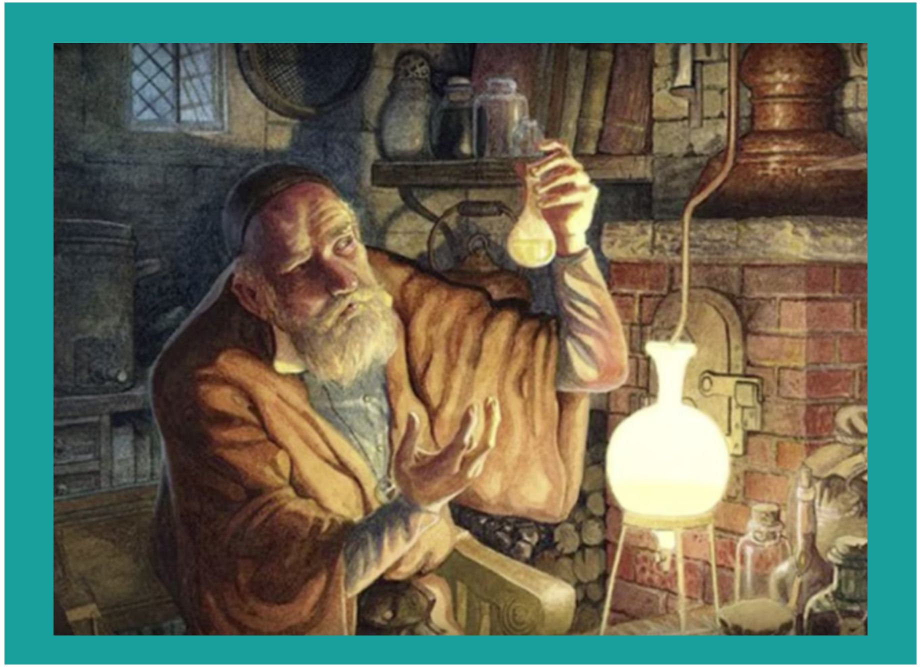


We tried a lot of architectures

- optimizers : **Adam**
- activations : **GELU**
- norm layers : **Batch**
- Dropout : **0.75**
- Weight decay : **1e-8**
- Stochastic weighting average
- **6 layers** MLPs

Results

Let the alchemy talk!



NIPS 2017: *Machine Learning has become alchemy*
https://www.youtube.com/watch?v=x7psGHgatGM

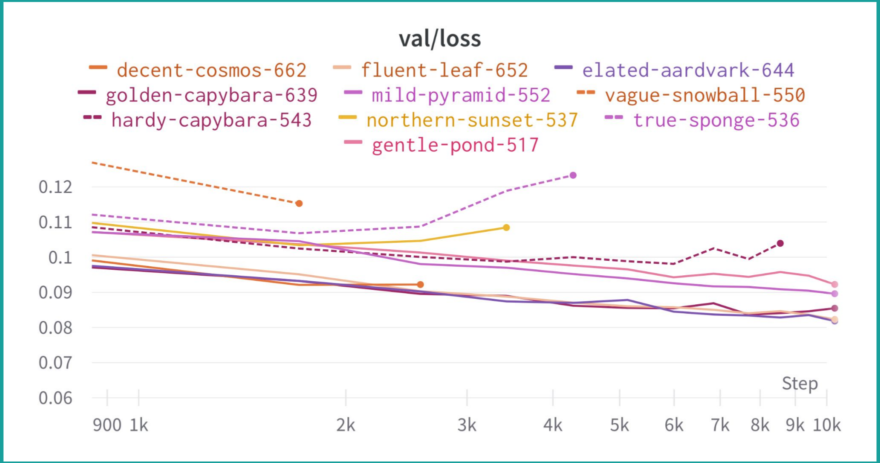
At training time:

Train the MLP (aka the **universal approximator**) for a binary classification task. For each pair of nodes:

1. Concatenate their features, aka *mix the ingredients*
2. Train the MLP using the *best tools*
3. Let the magic happen

Results

Let the alchemy act !



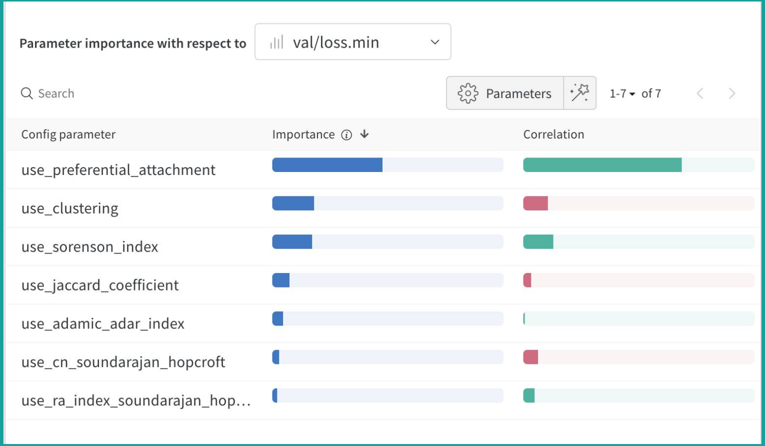
Wandb experiments

Main results:

Model	Public score on the test set
SPECTER + graph features + neighbors mean emb.	0.0988
SPECTER + graph features + neighbors mean emb. + keywords emb	0.0931
SPECTER + graph features + neighbors mean emb. + nb common keywords + intersection neighbors	**0.0822**
SPECTER + graph features + neighbors mean emb + nb common keywords + link prediction features + intersection neighbors	0.0839

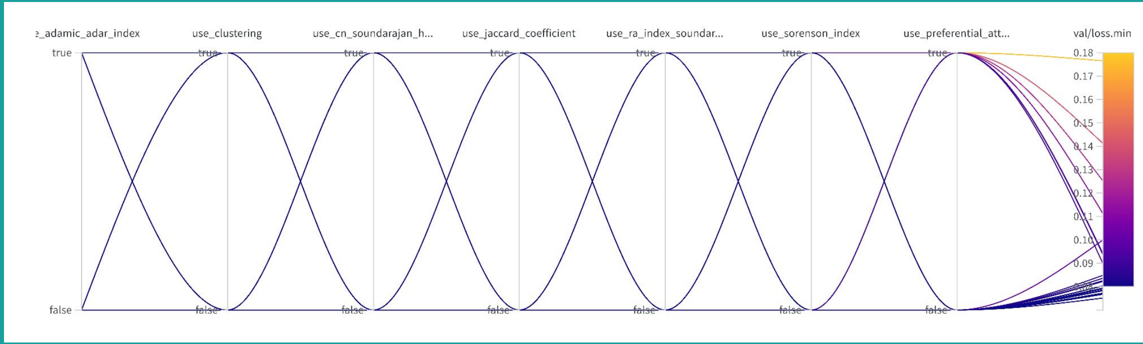
Results

Let the alchemy act !



Parameter importance with respect to val/loss.min

Config parameter	Importance	Correlation
use_preferential_attachment		
use_clustering		
use_sorenson_index		
use_jaccard_coefficient		
use_adamic_adar_index		
use_cn_soundarajan_hopcroft		
use_ra_index_soundarajan_hop…		



Discussions

What did worked and what did not?

We have successfully built a framework for citation prediction using text and graphs. We think that:

- Textual information seems to be very powerful (thanks to Language Models)
- Combining features is important to improve performances
- Handcrafted features help the model at a low computational cost

We also have various futur ideas:

- Should we use a wider MLP architecture?
- How can we use the features from the authors?
- Would node2vec embeddings catch more resilient features?
- Can we leverage attention to combine neighbor embeddings

Code available on github - Logs available on wandb

https://github.com/younesbelkada/altegrad_challenge

https://wandb.ai/altegrad-gnn-link-prediction/altegrad_challenge

THANKS FOR LISTENING !