Kodiak

07.09.2019

INTRODUCTORY TALK

Dylan Bourgeois ML Engineer Candidate

You can find these slides **dtsbourg.me/kodiak**.

About me

@dtsbourg
contact@dtsbourg.me
dtsbourg.me

Education

Student at EPFL (Lausanne, Switzerland)

B.Sc. in Microengineering

M.Sc. in Robotics

Mainly focused on Machine Learning

Experience

Intern at LHCb (CERN)

Working on designing ML methods for the LHCb Trigger Upgrade. [Bourgeois et al, 2018b, Hasse et al, 2018]

Latest

Masters Thesis at Stanford

Supervised by Prs. J. Leskovec (Stanford, SNAP) & P. Vandergheynst (EPFL, LTS2).

Working on *Learning representations of source code from structure & context.*

MSc. Thesis

Learning representations of source code from structure & context.

Work done under supervision by
Pr. Jure Leskoveč / Dr. Michele Catasta *(SNAP, Stanford)*
Pr. Pierre Vandergheynst / Michaël Defferrard *(LTS2, EPFL)*

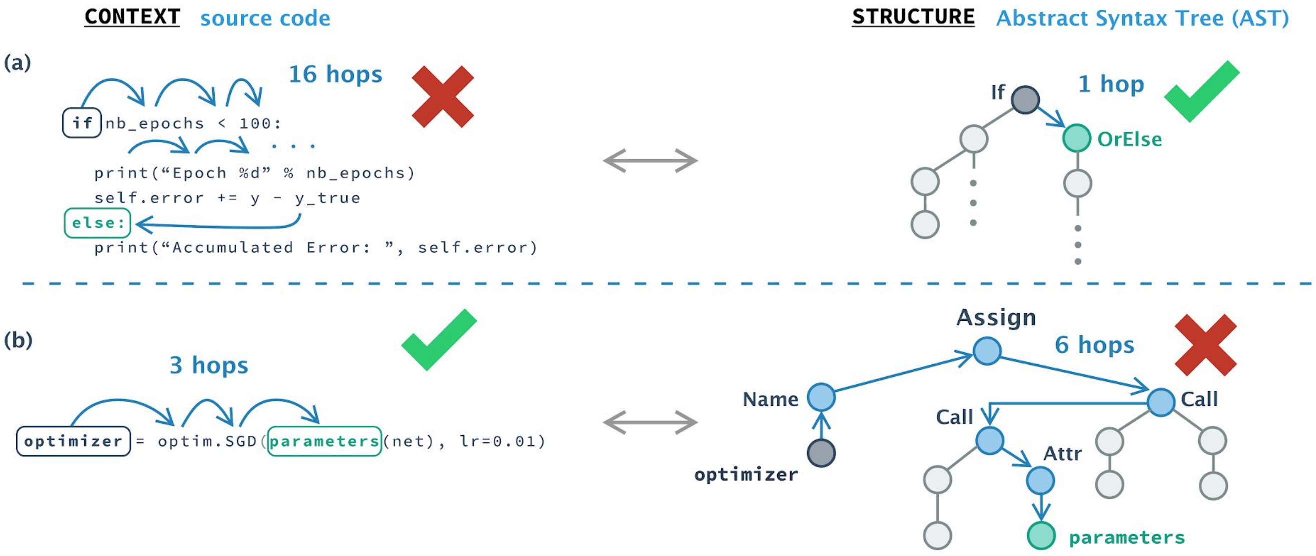
Capturing similarities of source code

Programming languages offer a unified interface, which is leveraged by programmers. The regularities in coding patterns can be used as a proxy for semantics.

Example applications

- Code recommendation
- Plagiarism detection
- Smarter development tools
- Error correction
- Smart search

Existing approaches only capture a single mode



Structure can efficiently model long-range dependencies which could fall outside the scope of the **context**.

Context can efficiently model co-occurences which can be overlooked by the **structure**.

We propose a hybrid approach

Our model leverages both **heuristics** and **regularities**, specifically through **structure**.

HEURISTICS (STRUCTURE)

We provide evidence for the importance of leveraging structure in the representation of source code.

REGULARITIES (CONTEXT)

We show that patterns in the input provide a decent signal.

HYBRID (OURS)

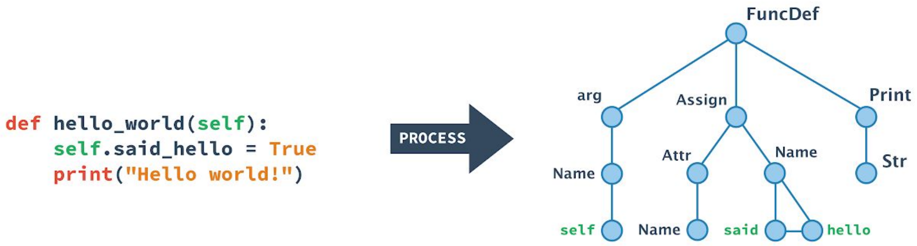
We propose a model which learns to recognize both structural and lexical patterns.

Structure is readily available for source code

Unlike natural language where parse trees are not unique and costly to infer.

HEURISTICS (STRUCTURE)

Structure can be extracted deterministically through compiler tools. It represents the language's grammar / syntax.



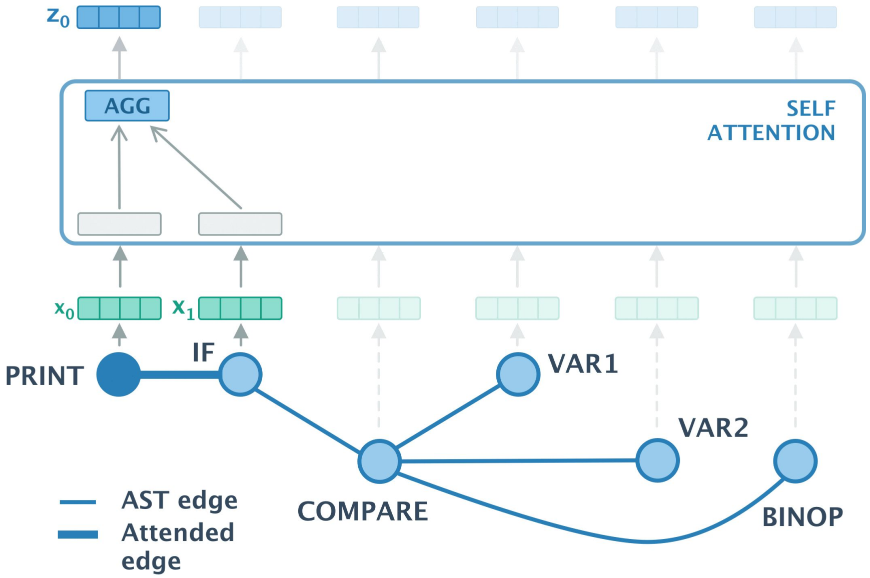
REGULARITIES (CONTEXT)

We show that patterns in the input provide a decent signal.

STRUCTURE

Graph Neural Networks capture local structure

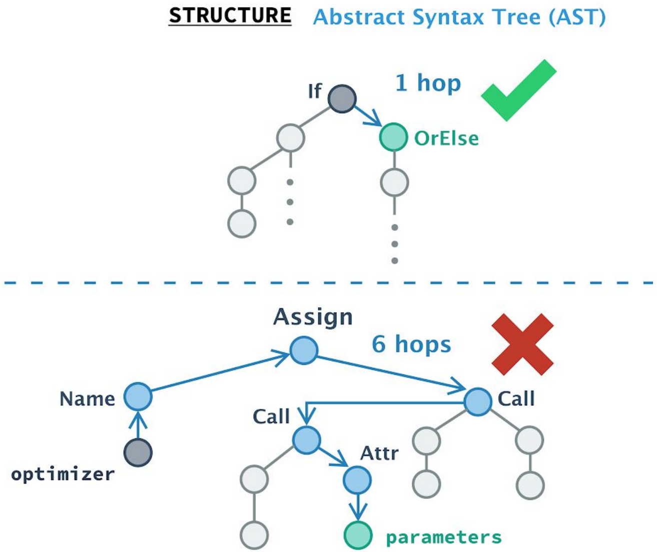
Node representations are computed from each of its neighbours.



STRUCTURE

Graph Neural Networks capture local structure

... but only in a limited receptive field!



Language Models are learned from context

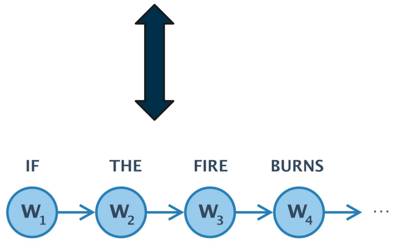
HEURISTICS (STRUCTURE)

Structure can be extracted deterministically through compiler tools. It represents the language's grammar / syntax.

REGULARITIES (CONTEXT)

Language models have been tried many times on Natural Language, but also on "Big Code". [Hindle et al., 2012]

$$P(w_1, \ldots, w_m) = \prod_{i=1}^{m} P(w_i \mid w_1, \ldots, w_{i-1}) \approx \prod_{i=1}^{m} P(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1})$$

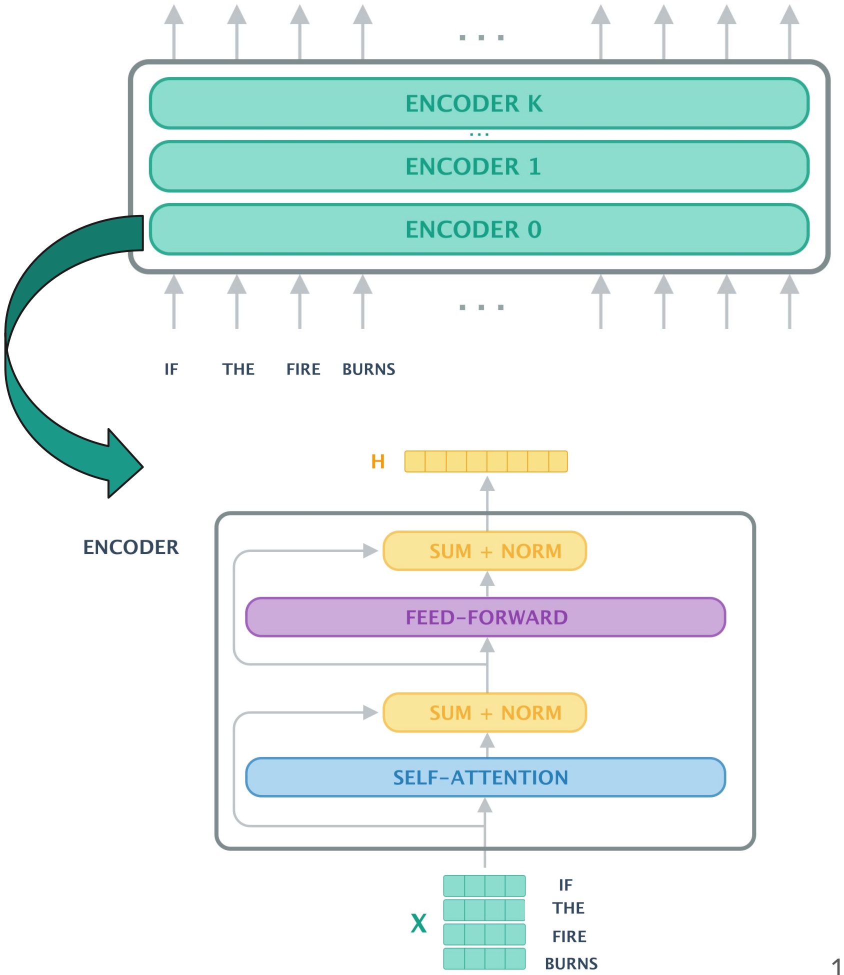


Linear Graphical Model
Markov Chain

CONTEXT

The Transformer is very good at capturing context

No assumptions are made on the underlying structure: the attention module can attend to all the elements in the sequence.

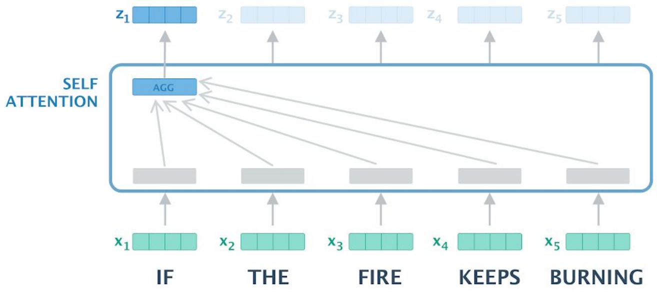


[Vaswani et al., 2017, Devlin et al, 2018]

INSIGHT

The Transformer is actually a GNN!

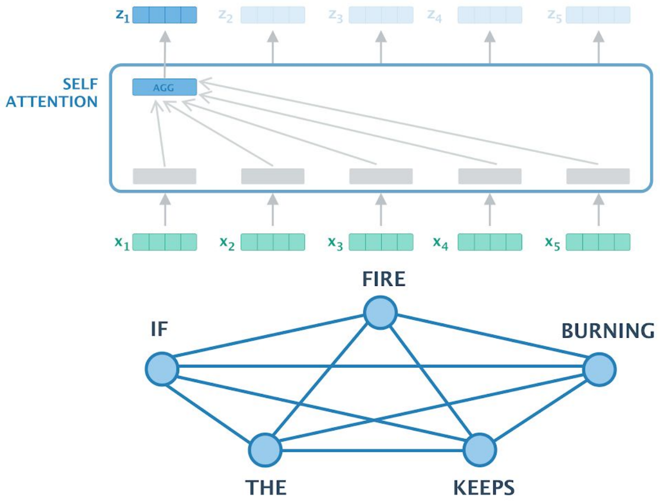
The encoder can be seen as a message-passing Graph Neural Network on a fully connected input graph.



INSIGHT

The Transformer is actually a GNN!

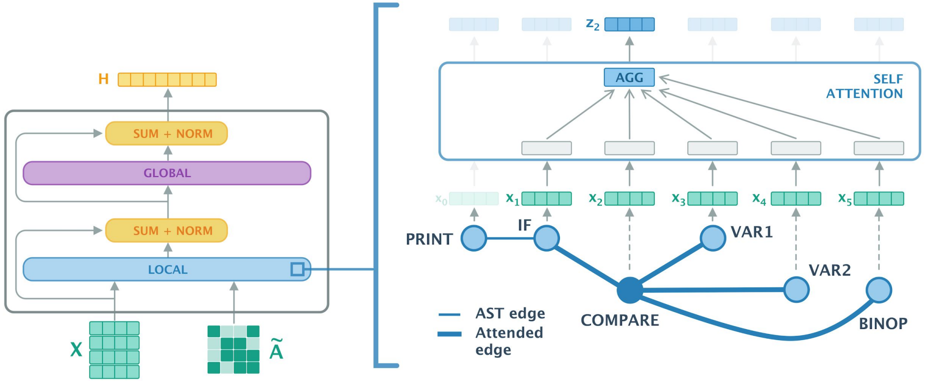
The encoder can be seen as a message-passing Graph Neural Network on a fully connected input graph.



OUR APPROACH - BiFocale

Capturing both local structure and global context.

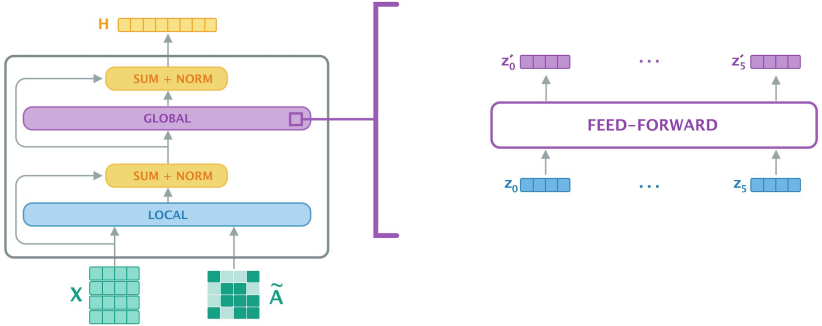
We can modify a Transformer encoder block to run on arbitrarily structured inputs.



OUR APPROACH - BiFocale

Capturing both local structure and global context.

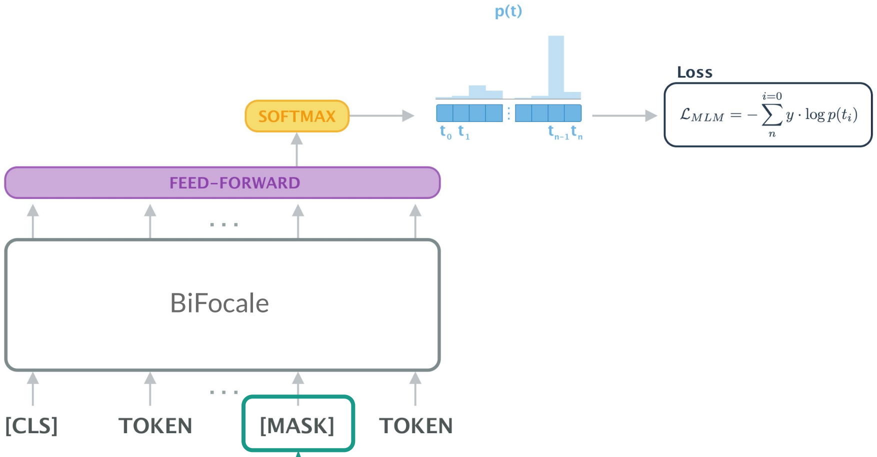
For example, with the masked attention formulation, we can modify a Transformer encoder block to run on arbitrarily structured inputs.



This hybrid model can also be pre-trained!

[Devlin et al, 2018; Radford et al., 2019]

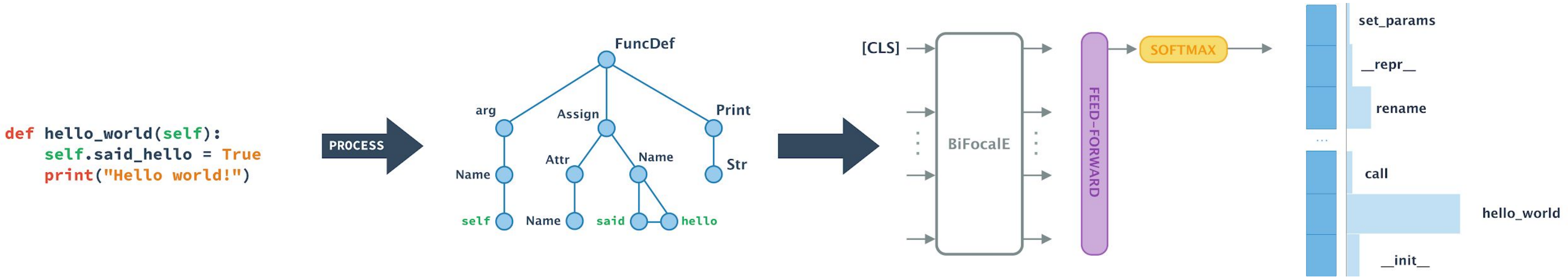
This has been shown to equip the model with an initial knowledge of the domain at hand. This solution should be closer to future relevant tasks in the hypothesis space.



Randomly masked tokens are hidden and predicted.

The model is fine-tuned to predict method names

Given a method definition, the model must predict a relevant name from ~10k.



… and is pretty good at it!

	BiFocalE		Alon'19 [9]		Alon'18 [8]	Fernandes'18 [19]	Allamanis'18 [7]
	Acc.	F1	Acc.	F1	Acc.	F1	Acc
Java *	**0.756**	**69.1**	0.633	59.5	0.473	51.4	—
Python *	**0.760**	**60.5**	—	—	0.511 @7	—	0.416

Trained on

*** 1M LoC** *** 3M LoC**

Predictions hint at the model's hybrid nature

CORRECT PREDICTIONS

1

```
def get_config(self):
        return {
            'mean': self.mean,
            'stddev': self.stddev,
            'seed': self.seed
        }
```

Predictions
```
0. get_config (1.0)
1. _updated_config (0.0)
2. _preprocess_conv3d_kernel (0.0)
```

2

```
def __init__(self, minval=0, maxval=5,
seed=None):
        self.minval = minval
        self.maxval = maxval
        self.seed = seed
```

Predictions
```
0. __init__ (1.0)
1. on_train_begin (0.0)
2. preprocess_input (0.0)
```

INCORRECT PREDICTIONS

1

```
def glorot_normal(seed=None):
    return VarianceScaling(scale=1.,
            mode='fan_avg',
            distribution='normal',
            seed=seed)
```

Predictions
```
0. he_normal (0.209)
1. lecun_normal (0.198)
2. lecun_uniform (0.198)
```

BiFocale is a hybrid!

The model can leverage both **co-occurrence based semantics** as well as structural similarities.

```
def sigmoid(x):
    return 1. / (1. + np.exp(-x))
```

Predictions

0. tanh (0.525)

```
def tanh(x):
    return np.tanh(x)
```

1. softplus (0.335)

```
def softplus(x):
    return np.log(1. + np.exp(x))
```

2. softsign (0.104)

```
def softsign(x):
    return x / (1 + np.abs(x))
```

BiFocale is a hybrid!

The model can leverage both co-occurrence based semantics as well as **structural similarities**.

```
def sigmoid(x):
    return 1. / (1. + np.exp(-x))
```

Predictions

0. tanh (0.525)

```
def tanh(x):
    return np.tanh(x)
```

1. softplus (0.335)

```
def softplus(x):
    return np.log(1. + np.exp(x))
```

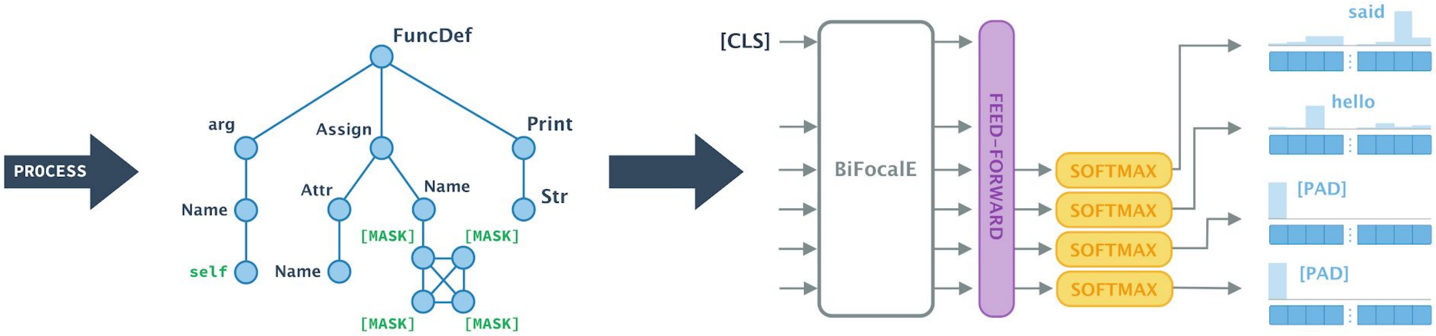
2. softsign (0.104)

```
def softsign(x):
    return x / (1 + np.abs(x))
```

The model is fine-tuned to predict variable names

We hide a variable from a snippet of code and ask the model to predict the masked variable's sub-tokens.

```python
def hello_world(self):
    self.said_hello = True
    print("Hello world!")
```



BiFocale sets a new SoTA!

We also show how heuristics are useful for learning on structured grammars.

	Accuracy			
	@1	@3	@5	@7
BERT	0.3	0.43	0.48	0.52
BIFOCALE	**0.59**	**0.792**	**0.833**	**0.849**
Alon et al.'18 [8]	—	—	—	0.567
Allamanis et al.'18 [7]	0.323	0.408	0.437	—

Predictions hint at the model’s hybrid nature

CORRECT PREDICTIONS

1
```
for cell in self.cells:
  if isinstance(cell, Layer):
    trainable_weights += cell.trainable_weights
```

Predictions

['cell', '[PAD]', '[PAD]', '[PAD]']

2
```
def __call__(self, shape, dtype=None):
        return K.constant(0, shape=shape,
               dtype=dtype)
```

Predictions

['self', '[PAD]', '[PAD]', '[PAD]']

INCORRECT PREDICTIONS

1
```
def top_k_categorical_accuracy(y_true,
y_pred, k=5):
    return K.mean(K.in_top_k(y_pred,
                  K.argmax(y_true, axis=-1), k),
                  axis=-1)
```

Predictions

0. ['y', 'true', 'true', 'true']
1. ['self', '[PAD]', '[PAD]', '[PAD]']
2. ['true', 'train', 'train', 'train']

For more ...

@dtsbourg
contact@dtsbourg.me
dtsbourg.me

Stanford

GNN Explainer

Interpretability methods for Graph Neural Networks. [Ying et al, 2019]

CERN

LHCb Trigger Upgrade (CERN)

Working on designing ML methods for the LHCb Trigger Upgrade. [Bourgeois et al, 2018b, Hasse et al, 2018]

EPFL

Media Observatory

Monitoring the media ecosystem. [Bourgeois et al, 2018a, Rappaz et al, 2019]

CERN

LHCb Trigger Upgrade (CERN)

Working on designing ML methods for the LHCb Trigger Upgrade. [Bourgeois et al, 2018b, Hasse et al, 2018]

EPFL

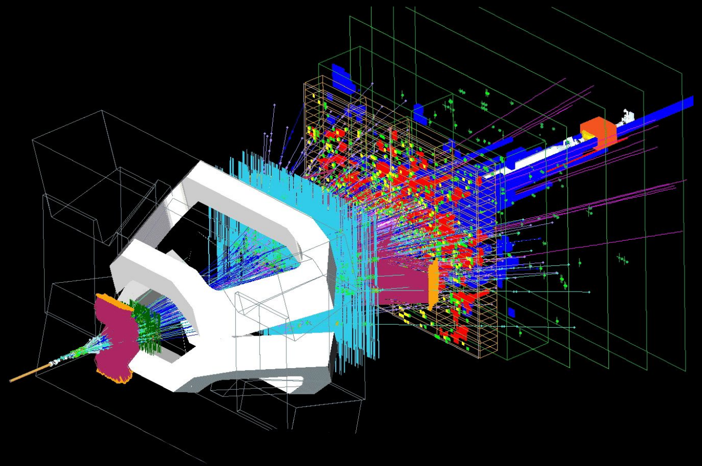
Media Observatory

Monitoring the media ecosystem. [Bourgeois et al, 2018a, Rappaz et al, 2019]

[Bourgeois et al, 2018b, Hasse et al, 2018]

Fast selection of interesting collisions

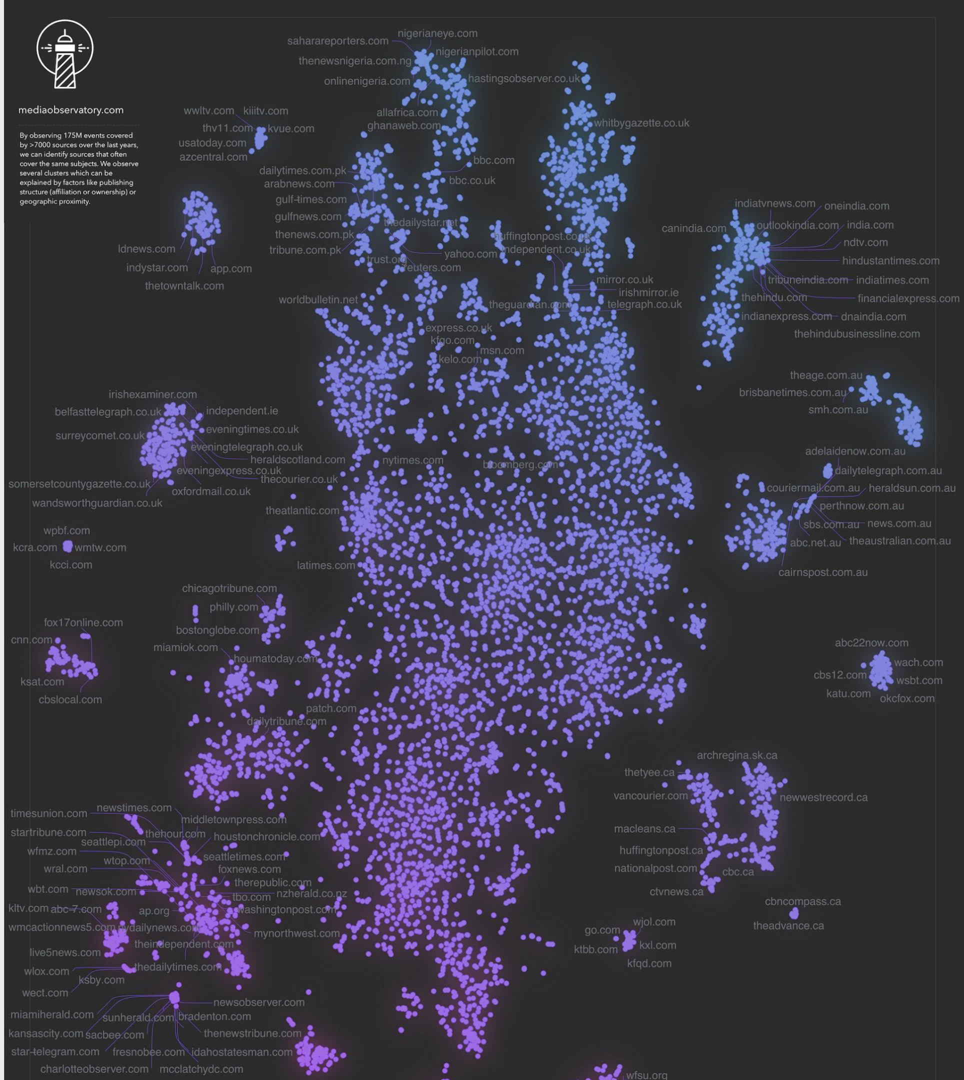
The experiment would be throttled if all events were saved to disk so the Trigger acts as a filter.



[Bourgeois et al, 2018a, Rappaz et al, 2019]

Studying coverage patterns to observe the news ecosystem

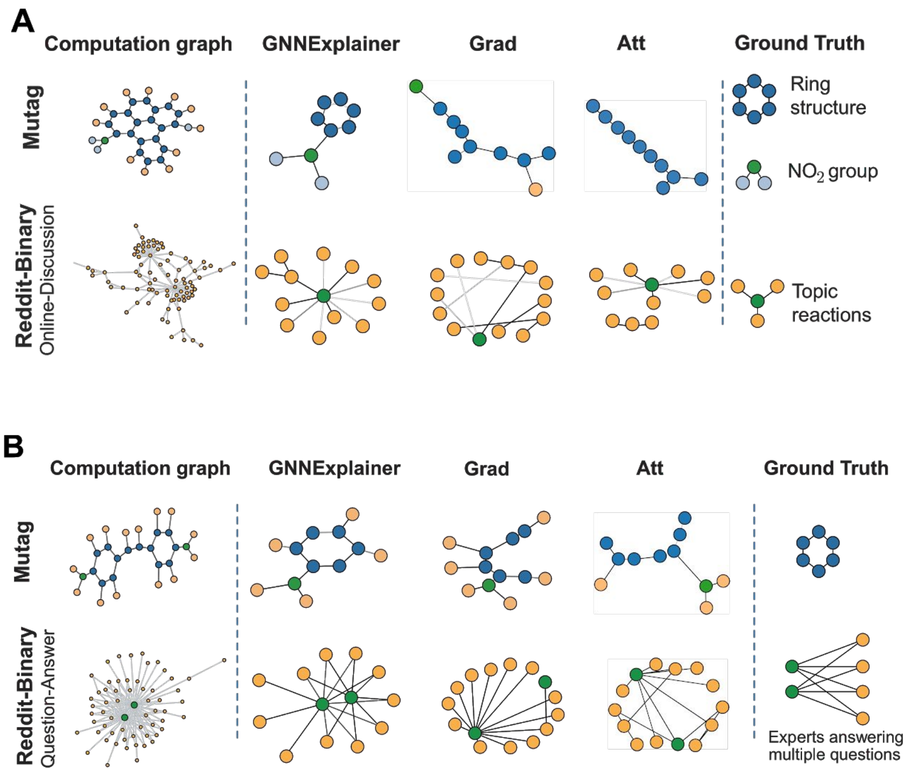
We build a map of source similarity without relying on a ground truth to estimate bias.



[Ying et al, 2019]

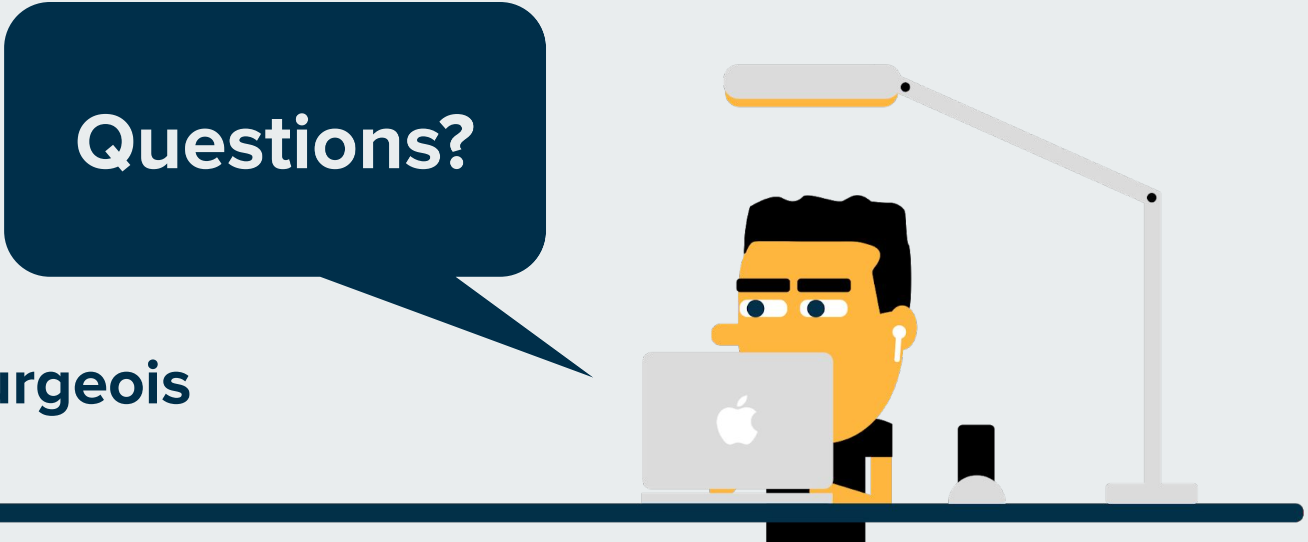
Interpretability methods for Graph Neural Networks

Works for any trained GNN, showing important structural and feature-based information that is most relevant to a prediction.



You can find these slides **dtsbourg.me/kodiak**.

Thank you ~~K~~odiak!



Dylan Bourgeois
@dtsbourg

Bibliography - My work

[Ying et al, 2019] *GNN Explainer: A Tool for Post-hoc Explanation of Graph Neural Networks* *
R. Ying, D. Bourgeois, J. You, M. Zitnik, J. Leskovec

[Bourgeois et al, 2018a] *Selection Bias in News Coverage: Learning It, Fighting It*
D. Bourgeois, J. Rappaz, K. Aberer, WWW'18

[Rappaz et al, 2019] *A Dynamic Embedding Model of the Media Landscape*
J. Rappaz, D. Bourgeois, K. Aberer, WWW'19

[Bourgeois et al, 2018b] *Using holistic information in the Trigger*
D. Bourgeois, C. Fitzpatrick, S. Stahl, LHCb Pub

[Hasse et al, 2018] *New approaches for track reconstruction in LHCb's Vertex Locator*
C. Hasse, J. Albrecht, B. Couturier, D. Bourgeois, V. Coco, N. Nolte, S. Ponce, JHEP'18

* Submitted to NeurIPS under the title *GNN Explainer: Generating Explanations for Graph Neural Networks*.

Bibliography

[Allamanis, 2018] Allamanis, M. (2018). *The adverse effects of code duplication in machine learning models of code.* arxiv:1812.06469.

[Allamanis et al., 2015] Allamanis, M., Barr, E. T., Bird, C., and Sutton, C. (2015). *Suggesting accurate method and class names*. ESEC/FSE 2015, pages 38–49

[Allamanis et al., 2018a] Allamanis, M., Barr, E. T., Devanbu, P. T., and Sutton, C. A. (2018a). *A survey of machine learning for big code and naturalness*. ACM Comput. Surv., 51:81:1–81:37.

[Allamanis et al., 2018b] Allamanis, M., Brockschmidt, M., and Khademi, M. (2018b). *Learning to represent programs with graphs.* ICLR.

[Alon et al., 2018] U. Alon, M. Zilberstein, O. Levy, and E. Yahav. *A general path-based representation for predicting program properties.* PLDI 2018.

[Alon et al., 2019] Alon, U., Zilberstein, M., Levy, O., and Yahav, E. (2019). *Code2vec: Learning distributed representations of code*. POPL.

Bibliography

[Bengio et al., 2003] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). *A neural probabilistic language model*. J. Mach. Learn. Res., 3:1137–1155.

[Collobert and Weston, 2008] Collobert, R. and Weston, J. (2008). *A unified architecture for natural language processing: Deep neural networks with multitask learning.* ICML '08.

[Deerwester et al.,1990] Deerwester,S.C.,Dumais,S.T.,Landauer,T.K.,Furnas,G.W.,and Harshman, R. A. (1990). *Indexing by latent semantic analysis.* JASIS, 41:391–407.

[Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). *BERT: Pre-training of deep bidirectional transformers for language understanding*. Arxiv:1810.04805.

[Firth,1957] Firth,J.R.(1957). *A synopsis of linguistic theory* 1930-55. Studies in Linguistic Analysis (special volume of the Philological Society), 1952-59:1–32.

[Fernandes,2018] P. Fernandes, M. Allamanis, and M. Brockschmidt. *Structured neural summarization*, 2018.

[Hindle et al., 2012] Hindle, A., Barr, E. T., Su, Z., Gabel, M., and Devanbu, P. (2012). *On the naturalness of software*. In ICSE '12, pages 837–847, IEEE Press.

Bibliography

[Hamilton et al., 2017] William L. Hamilton, Rex Ying, Jure Leskovec. Inductive Representation Learning on Large Graphs. NeurIPS 2017

[Mikolov et al., 2013] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). *Efficient estimation of word representations in vector space*. ICLR'13

[Radford et al., 2018] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). *Improving language understanding by generative pre-training.* OpenAI.

[Shannon, 1950] Shannon, C. (1950). *Prediction and entropy of printed english*. Bell Systems Technical Journal.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). *Attention is all you need.* In NeurIPS.

[Xu et al., 2019] Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019). *How powerful are graph neural networks?* In ICLR'19.