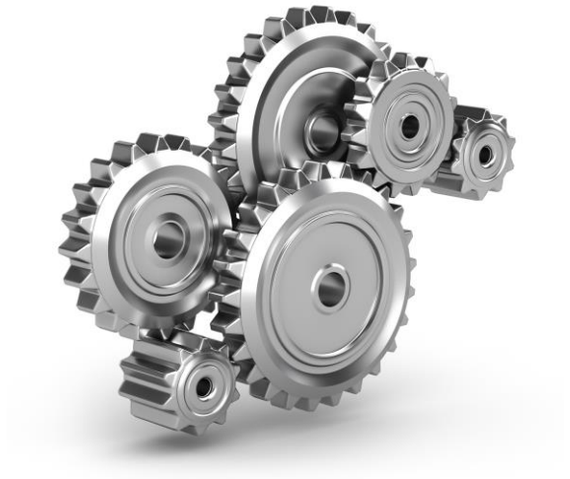
Learning from Source Code Identifiers

Miltos Allamanis

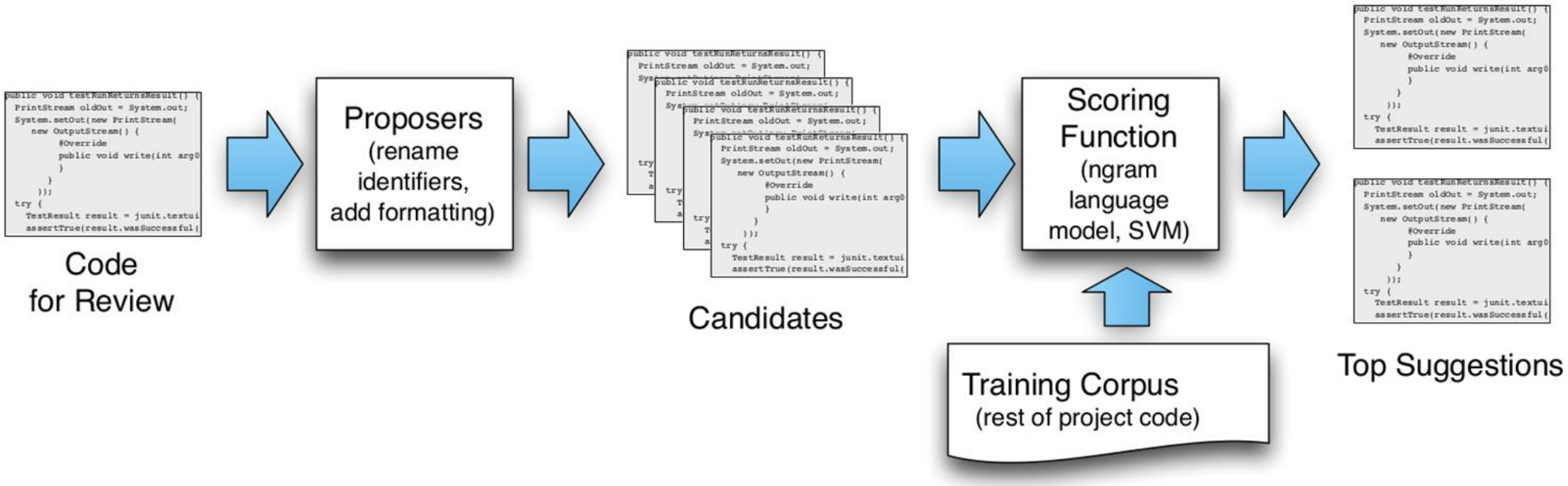
Learning to Name Source Code



A name reflects important aspects of code **functionality**.

Learning to name source code is a first step in *understanding* code through machine learning.

Variable Naming Conventions




```
ForkJoinTask<?> job;
if (task instanceof ForkJoinTask<?>) // avoid re-wrap
    job = (ForkJoinTask<?>) task;
else
    job = new
ForkJoinTask.AdaptedRunnableAction(task);
externalPush(job);
```

Suggestions

1. job (30%)
2. task (20%)
3. tsk (15%)

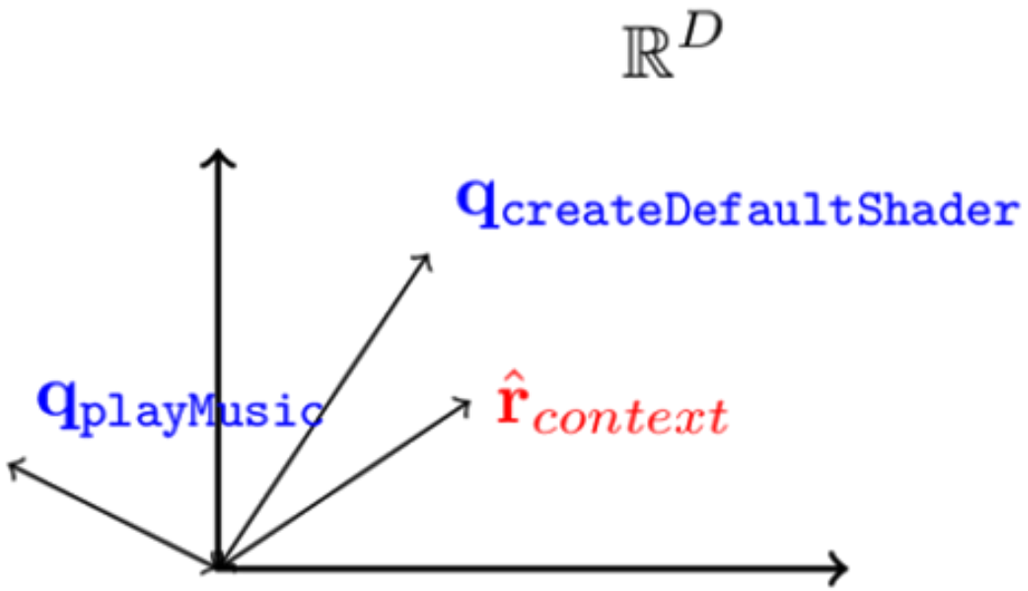
Allamanis *et al.* FSE 2014, 2015

A Machine Learning Model of Names

$$P(t|c) = \frac{exp(s_\theta(t,c))}{\sum_{t'} exp(s_\theta(t',c))}$$

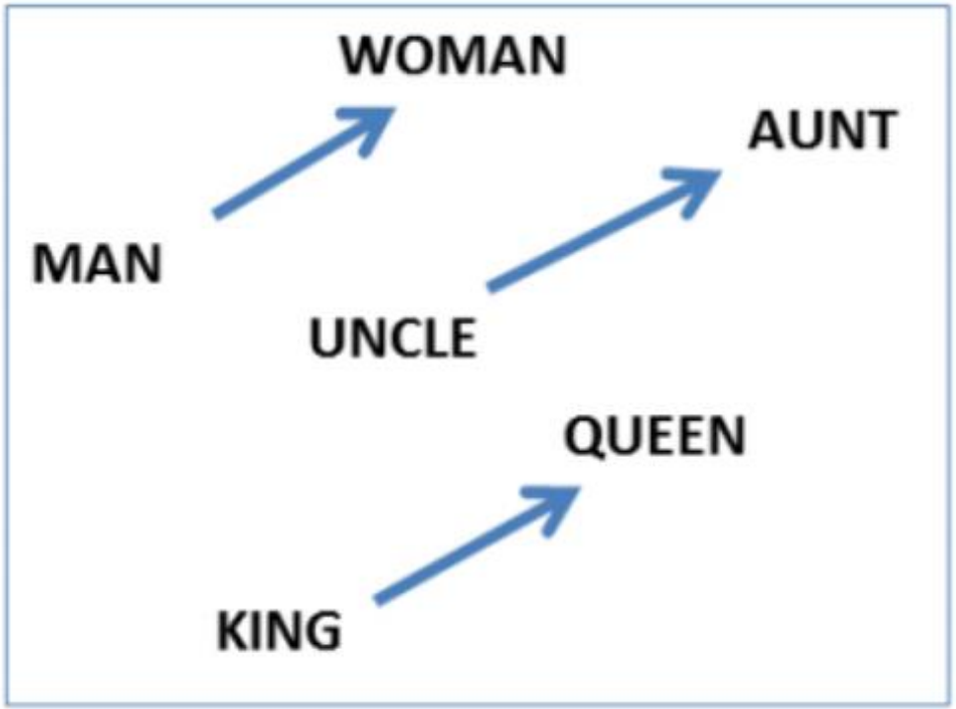
[Mnih, Hinton, 2007; Mnih, Teh, 2012; Mnih, Kavukcuoglu, 2013; Maddison, Tarlow, 2014]

Embedding Identifiers



$\mathbf{q}_t \in \mathbb{R}^D$ are "embeddings" ::: model parameters

[Mnih, Hinton, 2007; Mnih, Teh, 2012; Mnih, Kavukcuoglu, 2013; Maddison, Tarlow, 2014]



v(king) - v(man) + v(women) = v(queen)

Mikolov, Tomas, *et al.* "Efficient estimation of word representations in vector space." (2013).
https://code.google.com/p/word2vec/

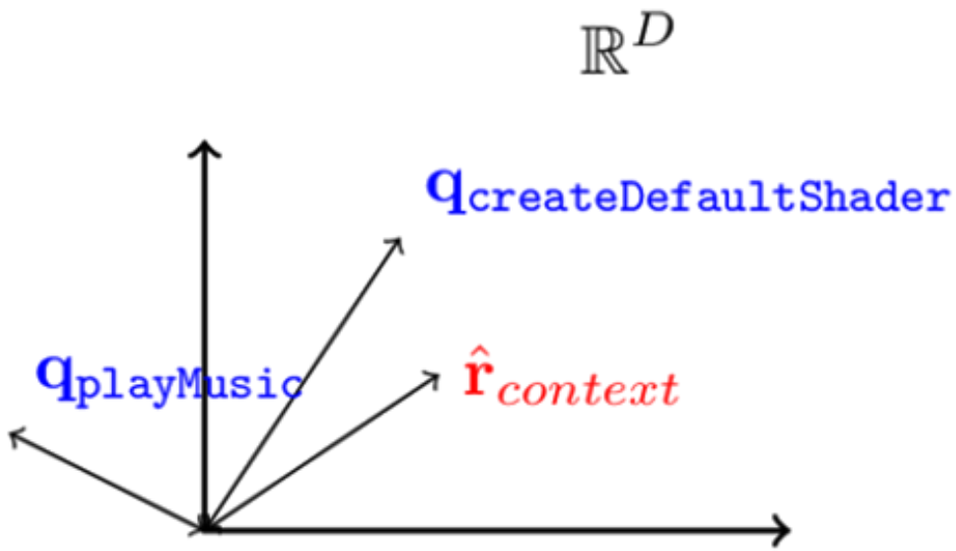
The Distributional Hypothesis in Linguistics

The ?????????? is walking

"You shall know a word by the company it keeps".

–John Rupert Firth, 1957

Embedding Identifiers

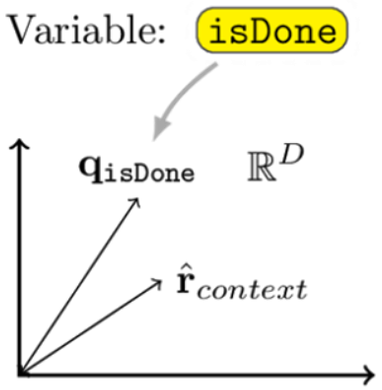


$$s_\theta(t, c) = \hat{\mathbf{r}}_{context}^\top q_t + b_t$$

$$P(t|c) = \frac{exp(s_\theta(t,c))}{\sum_{t'} exp(s_\theta(t',c))}$$

[Mnih, Hinton, 2007; Mnih, Teh, 2012; Mnih, Kavukcuoglu, 2013; Maddison, Tarlow, 2014]

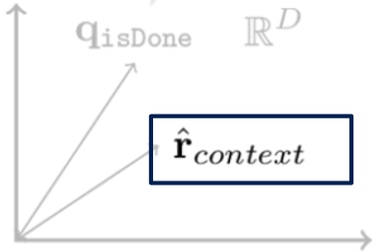
Neural Context Models of Source Code



$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

Neural Context Models of Source Code

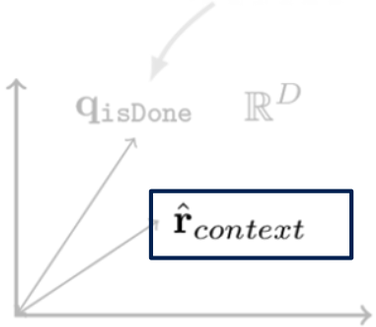
Variable: `isDone`



$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

Neural Context Models of Source Code

Variable: `isDone`



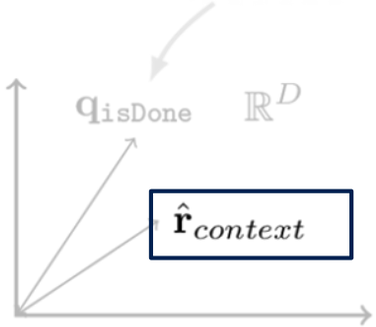
$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

Features:

`boolean` , `in:MethodBody` , `final`

Neural Context Models of Source Code

Variable: `isDone`



$$s_\theta(.) = \hat{\mathbf{r}}_{context}^\top \mathbf{q}_{\texttt{isDone}} + b_{\texttt{isDone}}$$

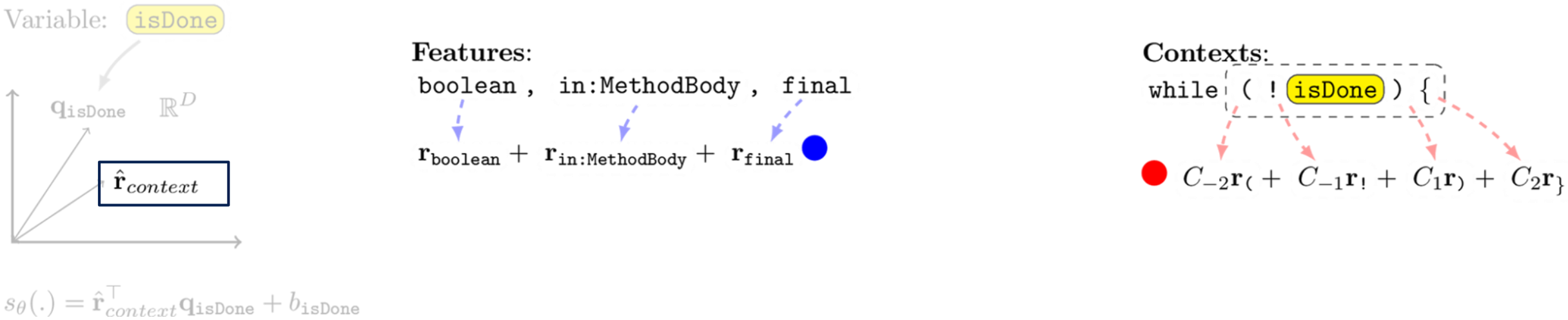
Features:

`boolean` , `in:MethodBody` , `final`

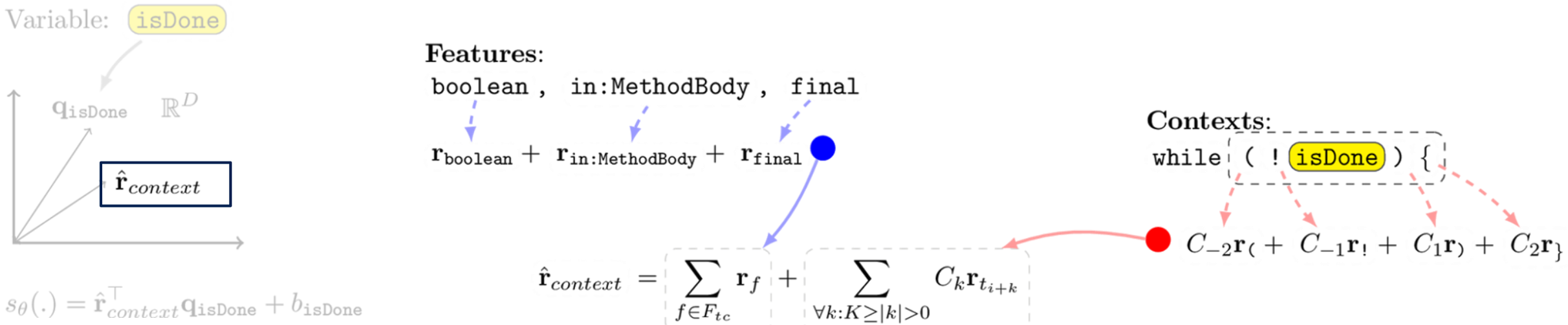
Contexts:

`while ( ! isDone ) {`

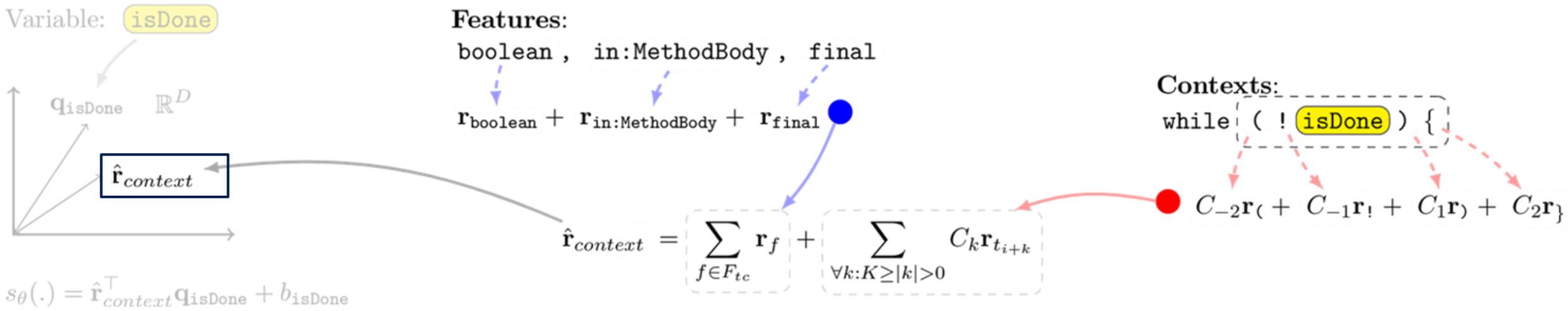
Neural Context Models of Source Code



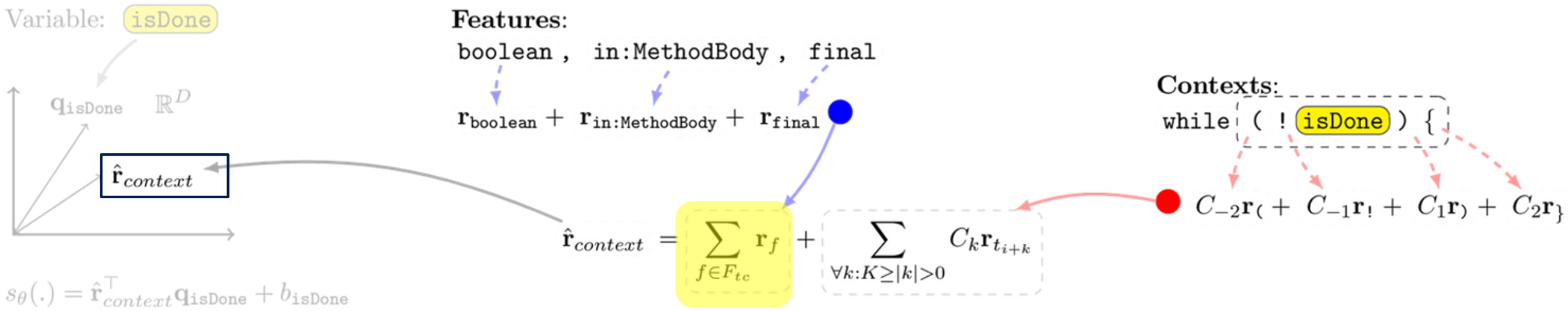
Neural Context Models of Source Code



Neural Context Models of Source Code

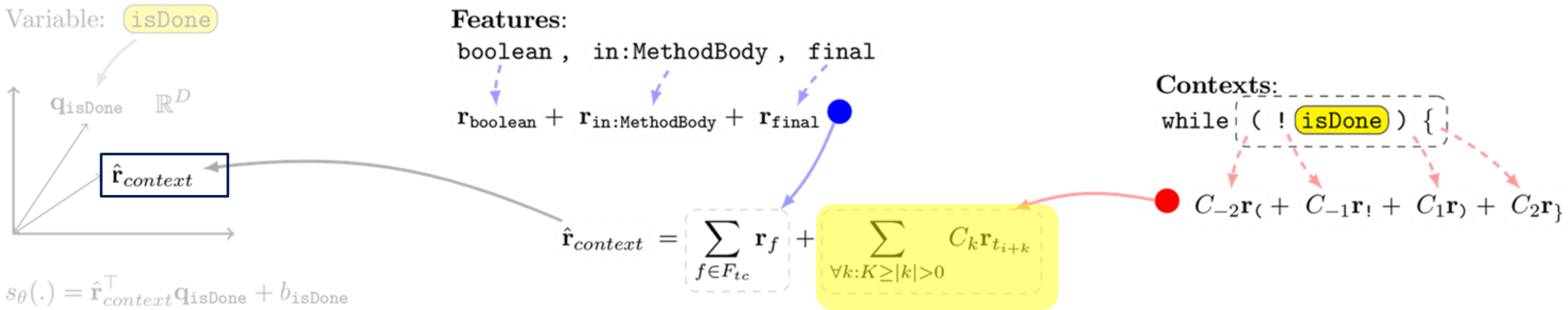


Neural Context Models of Source Code



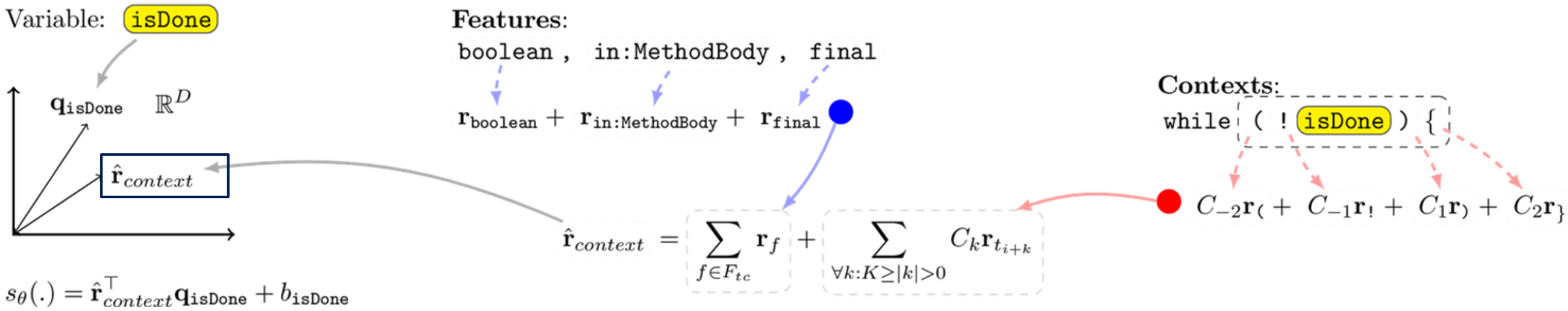
Global Information

Neural Context Models of Source Code

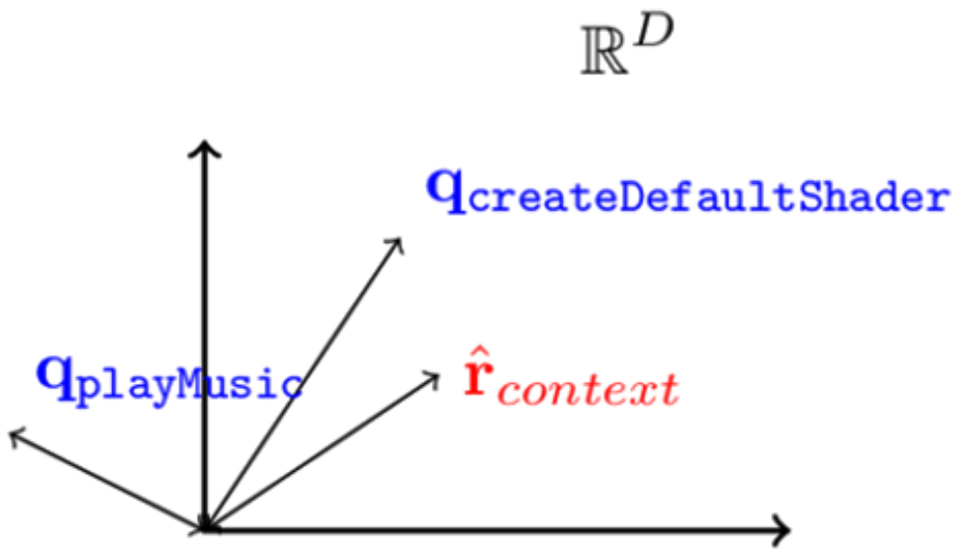


Local Information

Neural Context Models of Source Code



Embedding Identifiers



$$P(t|c) = \frac{exp(s_\theta(t,c))}{\sum_{t'} exp(s_\theta(t',c))}$$

[Mnih, Hinton, 2007; Mnih, Teh, 2012; Mnih, Kavukcuoglu, 2013; Maddison, Tarlow, 2014]

Neologisms

neologism

[nɪˈɒlədʒɪz(ə)m]

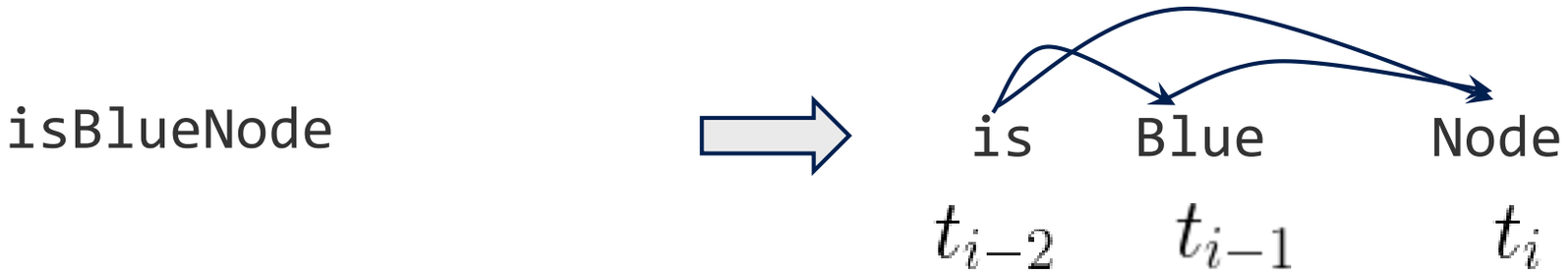
NOUN

a newly coined word or expression.

synonyms: new word · new expression · new term · new phrase · coinage · More

Powered by OxfordDictionaries · © Oxford University Press

Subtoken Context Models of Code



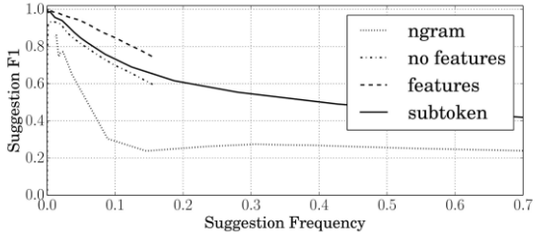
$$P\left(t_i | t_{i-1}, t_{i-2}, context\right)$$

Sequentially predict each subtoken given the context and the previous subtokens

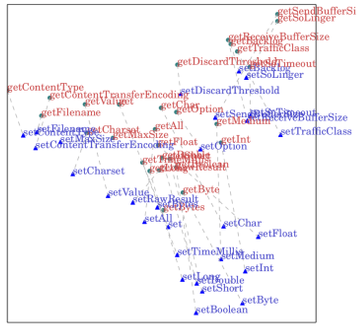
Training Data (project)

Train Neural Network

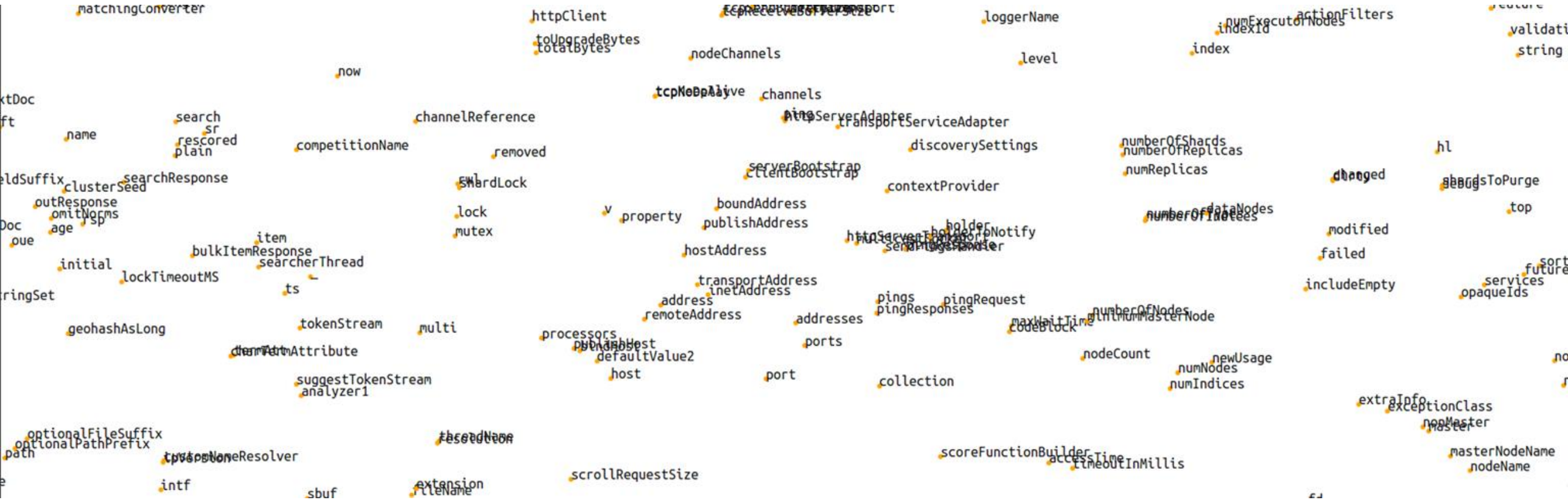
Suggest Names on Test Data



Embeddings



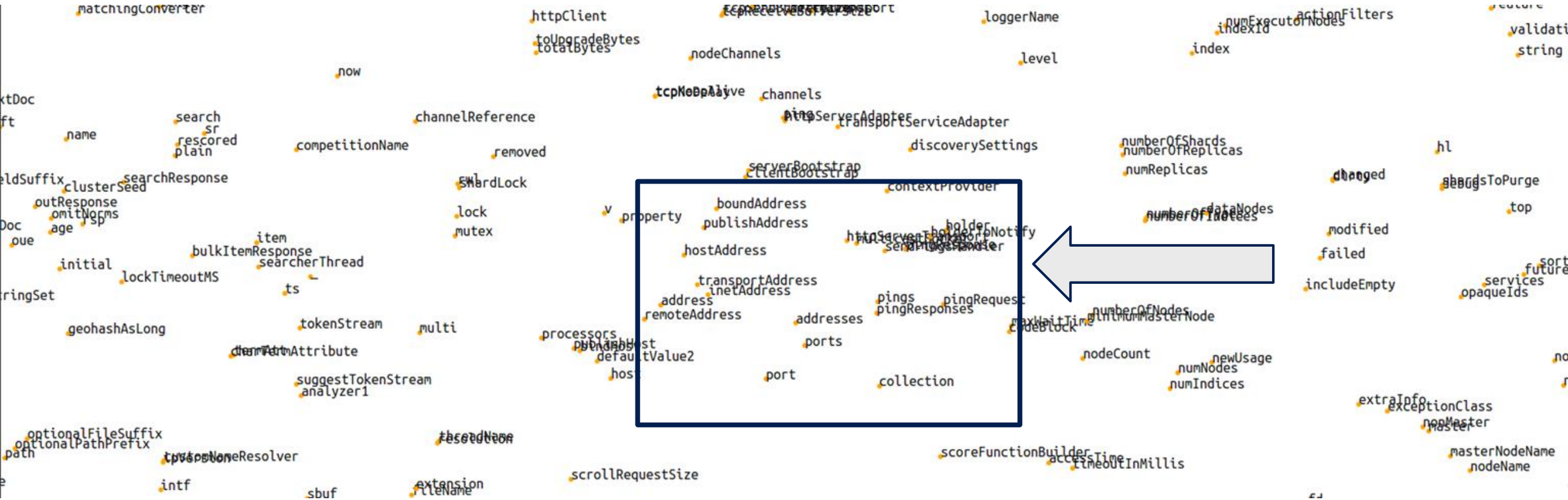
Embedding Visualization



T-SNE

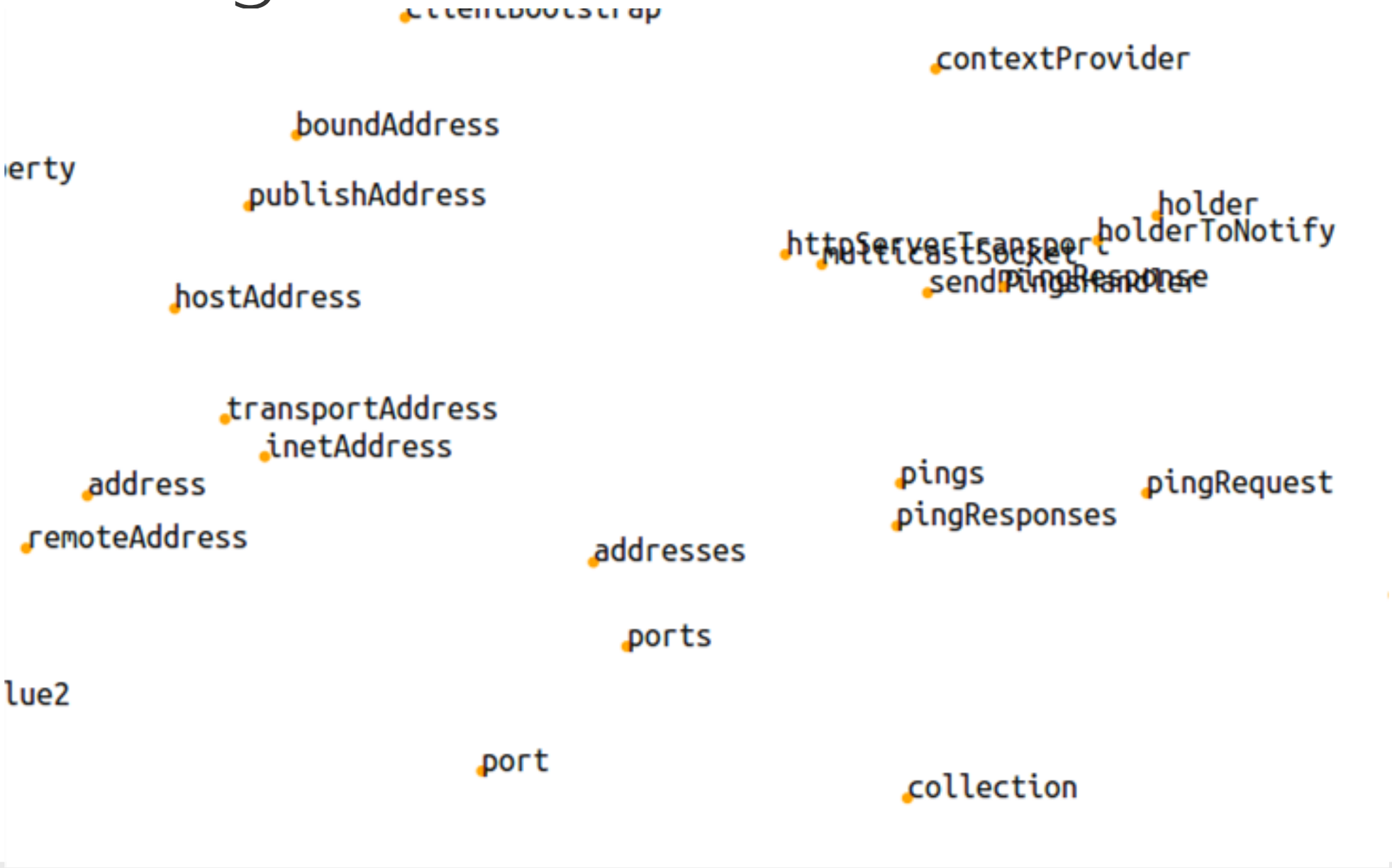
http://groups.inf.ed.ac.uk/cup/naturalize

Embedding Visualization



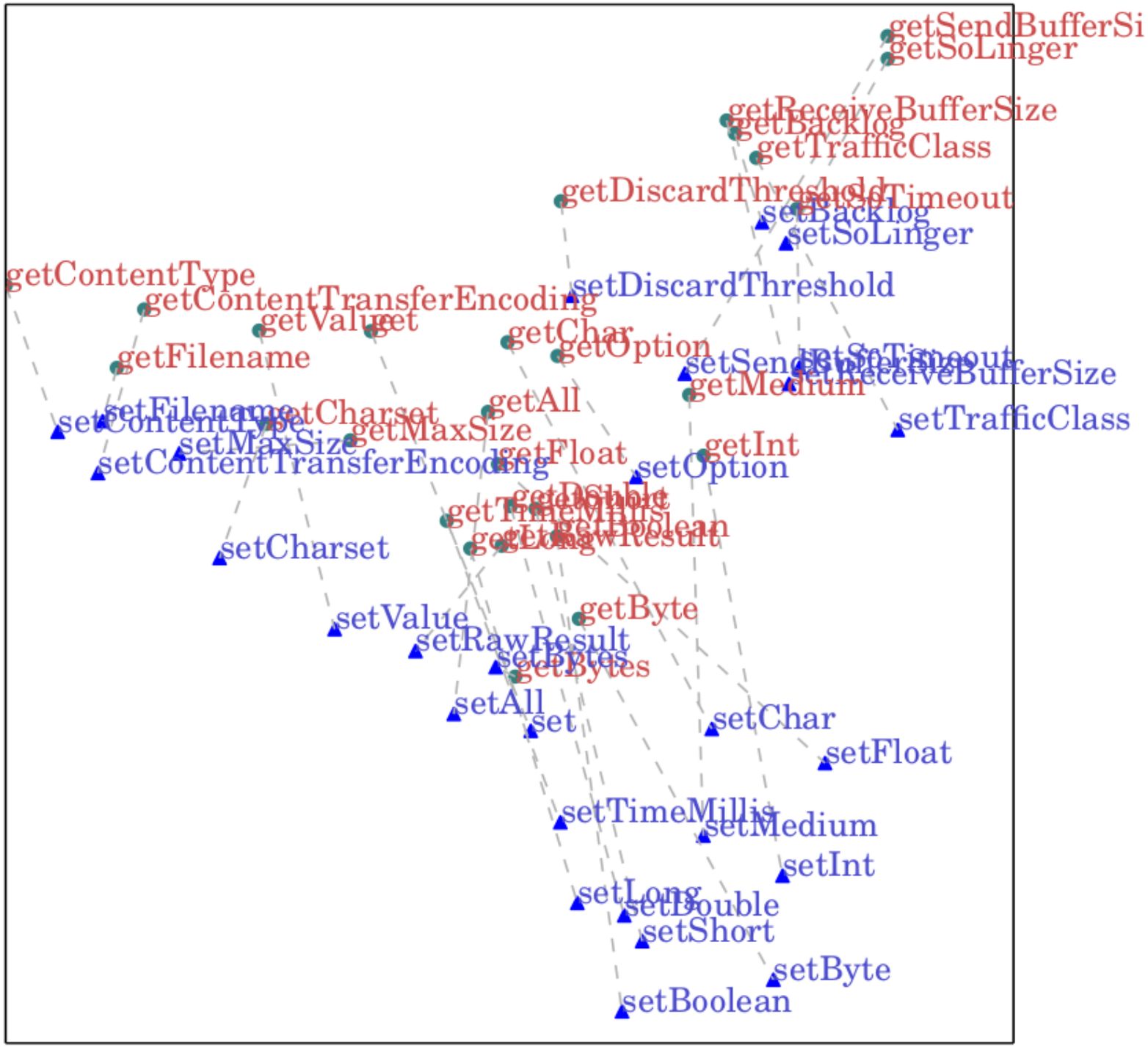
http://groups.inf.ed.ac.uk/cup/naturalize

Embedding Visualization

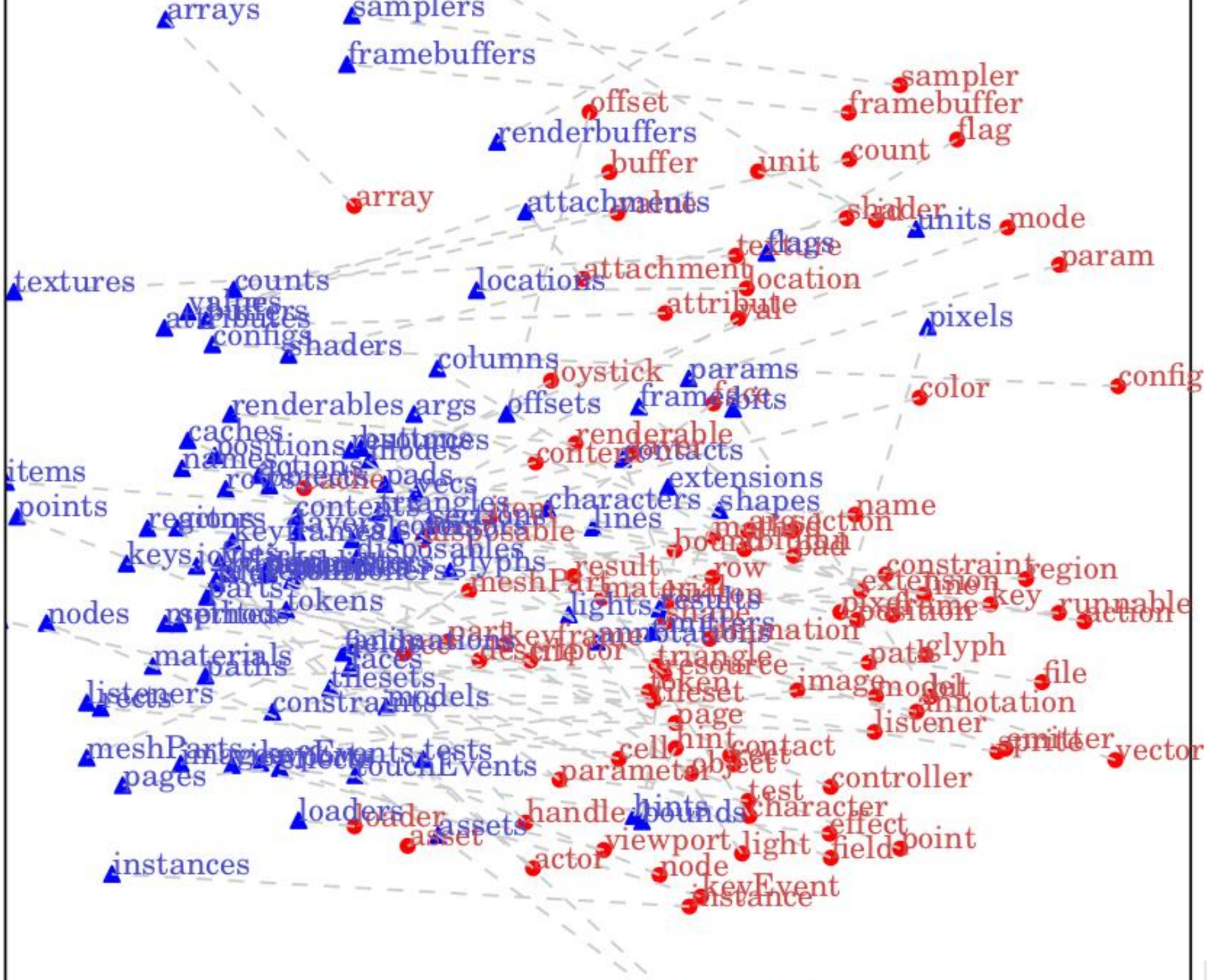


Embeddings - Closely Related Terms

Feature Model	Subtoken
camera – cam	6 – 5
padBottom – padLeft	Height – Width
dataOut – dataIn	swig – class
localAnchorA – localAnchorB	Min – Max
bodyA – bodyB	shape – collision
framebuffers – buffers	Left – Right
worldWidth – worldHeight	camera – cam
padRight – padLeft	TOUCH – KEY
jarg7 – jarg6_	end – start
spriteBatch – batch	loc – location



Setters vs Getters in bigbluebutton



Singular vs Plural in libgdx

Learning to Name Methods

Method Naming Problem

```
1 private void                                  () {
2   String vertexShader = "literal_1";
3   String fragmentShader = "literal_2";
4   shader = new ShaderProgram(vertexShader,
5       fragmentShader);
6   if(shader.isCompiled() == false)
7     throw new IllegalArgumentException(
8         "literal_3" + shader.getLog());
9 }
```

Method Naming Problem

```
1 private void                                  () {
2   String vertexShader = "literal_1";
3   String fragmentShader = "literal_2";
4   shader = new ShaderProgram(vertexShader,
5       fragmentShader);
6   if(shader.isCompiled() == false)
7     throw new IllegalArgumentException(
8         "literal_3" + shader.getLog());
9 }
```

Names describe what it ***does*** not what it ***is***

Models need to be “non-local”

Method Naming Problem

```
1 private void                                  () {
2   String vertexShader = "literal_1";
3   String fragmentShader = "literal_2";
4   shader = new ShaderProgram(vertexShader,
5       fragmentShader);
6   if(shader.isCompiled() == false)
7     throw new IllegalArgumentException(
8         "literal_3" + shader.getLog());
9 }
```

Suggestions:

- `create`
- `create?UNK?`
- `init`
- `createShader`

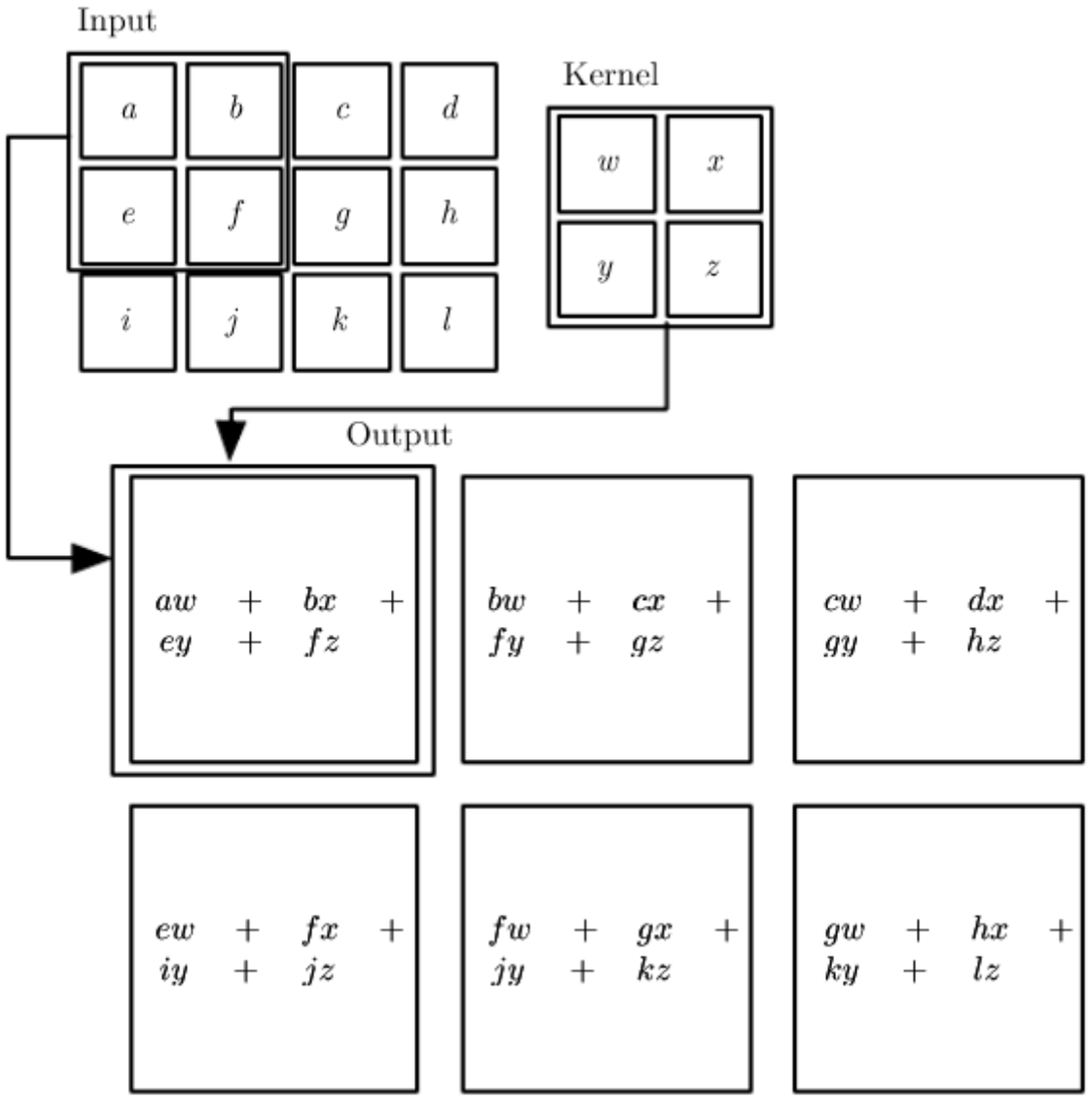
Method Naming Problem

```
1 private void createDefaultShader () {
2   String vertexShader = "literal_1";
3   String fragmentShader = "literal_2";
4   shader = new ShaderProgram(vertexShader,
5       fragmentShader);
6   if(shader.isCompiled() == false)
7     throw new IllegalArgumentException(
8         "literal_3" + shader.getLog());
9 }
```

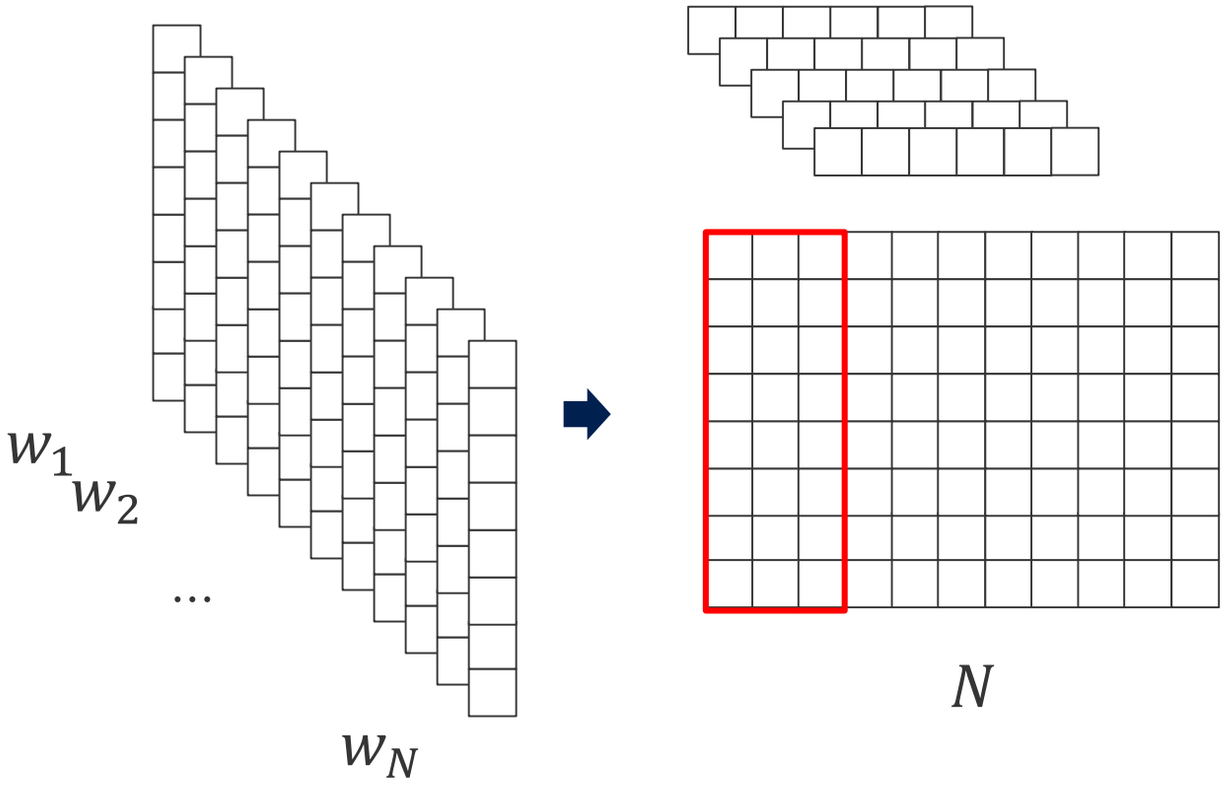
Suggestions:

- `create`
- `create?UNK?`
- `init`
- `createShader`

Convolution

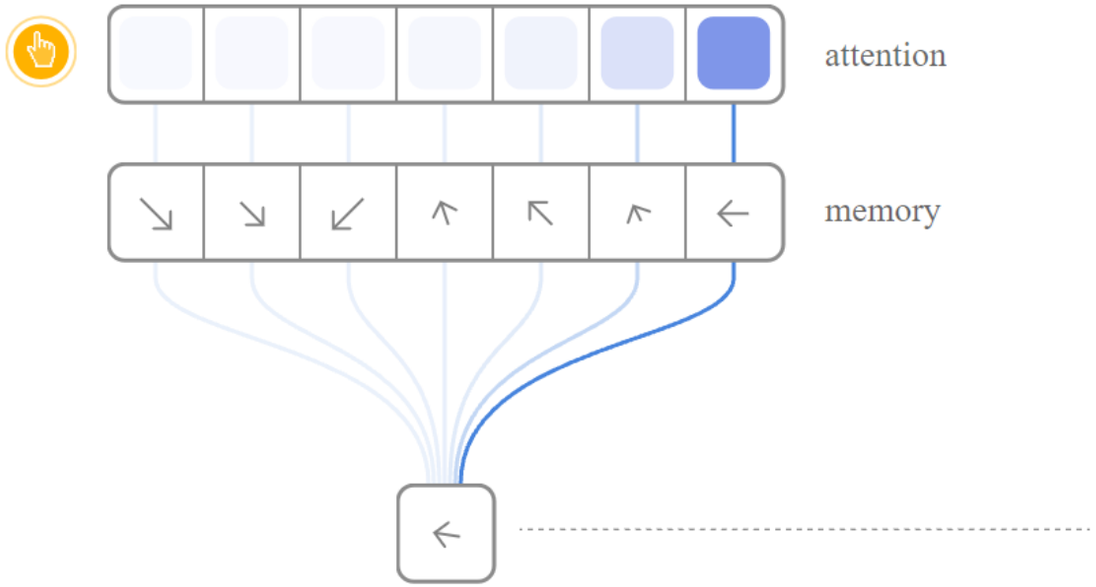


From deeplearningbook.org



1D Convolution

Attention Mechanisms in Deep Learning



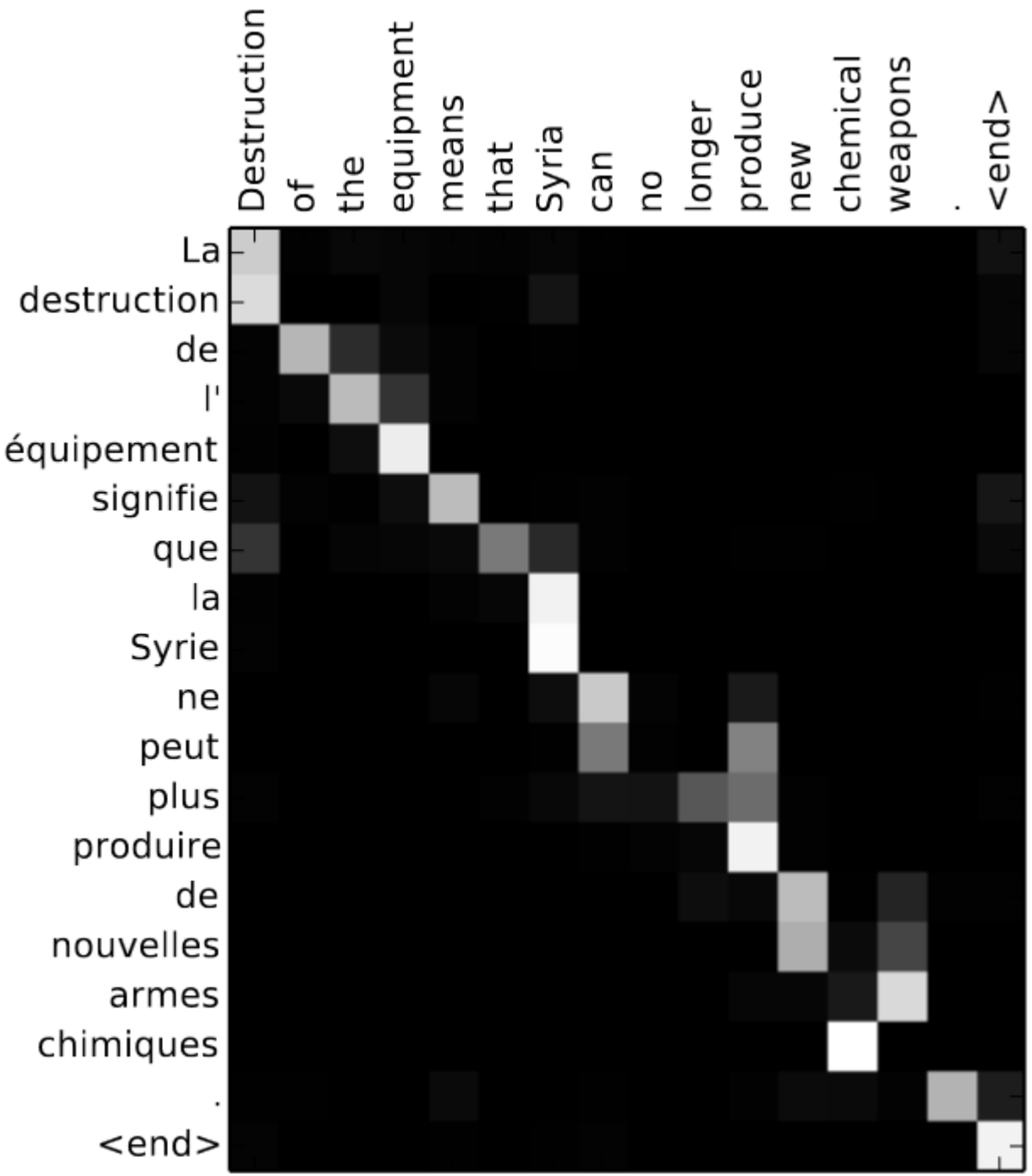
The RNN gives an attention distribution which describe how we spread out the amount we care about different memory positions

The read result is a weighted sum.

$$r \leftarrow \sum_i a_i M_i$$

http://distill.pub/2016/augmented-rnns/

Attention in Machine Translation



Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473* (2014).

Convolutional Attention Network for Naming Methods

Joint work with Hao Peng, Charles Sutton
ICML 2016

Neural Attention Models for Method Names

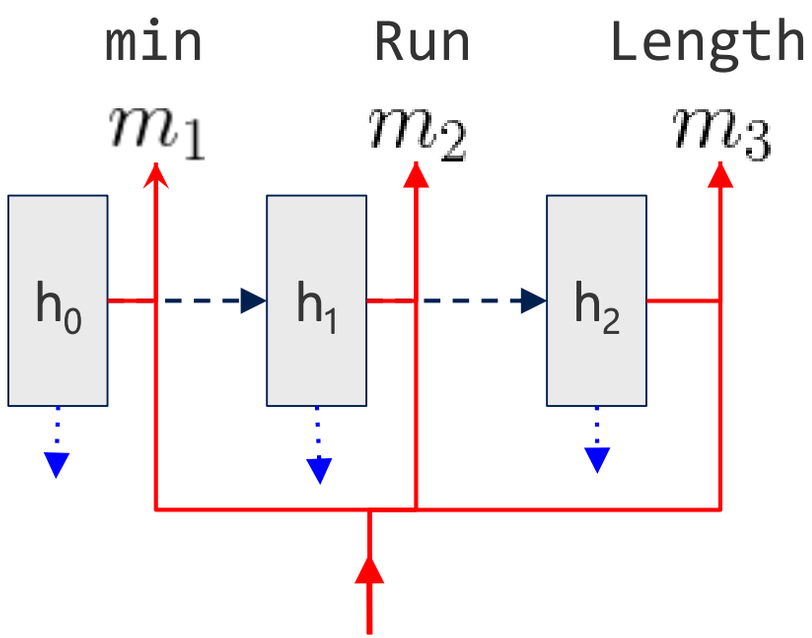
```
<s> { return payload != null ; } </s>
```

Method name starts with **`is`**

`isDragging()`

Architecture

(Subtoken) Summary



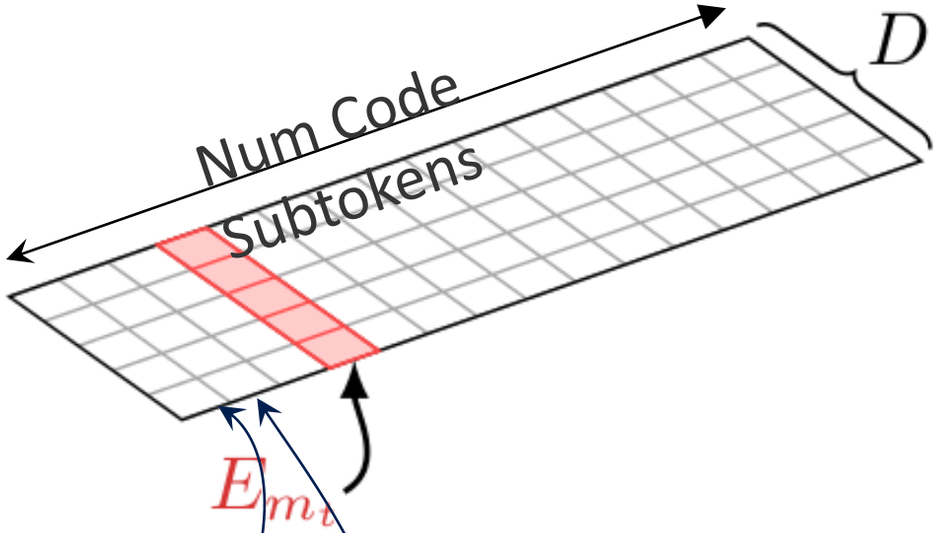
An RNN to predict summary subtokens

$$P(m_i | m_0 \ldots m_{i-1}, \mathbf{code})$$

Code

```
if (DEBUG) assert n >= 0;
int r = 0;
while (n >= MIN_MERGE) {
  r |= (n & 1);
  n >>= 1;
}
return n + r;
```

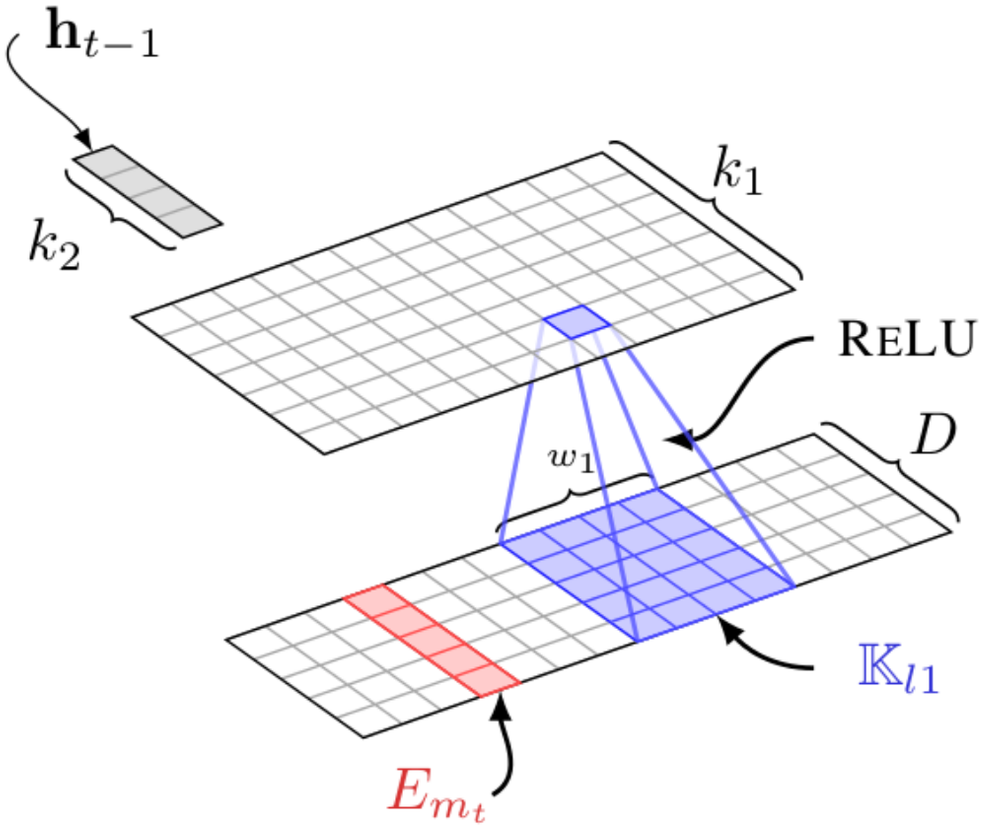
Context-Dependent Convolutional Attention Features



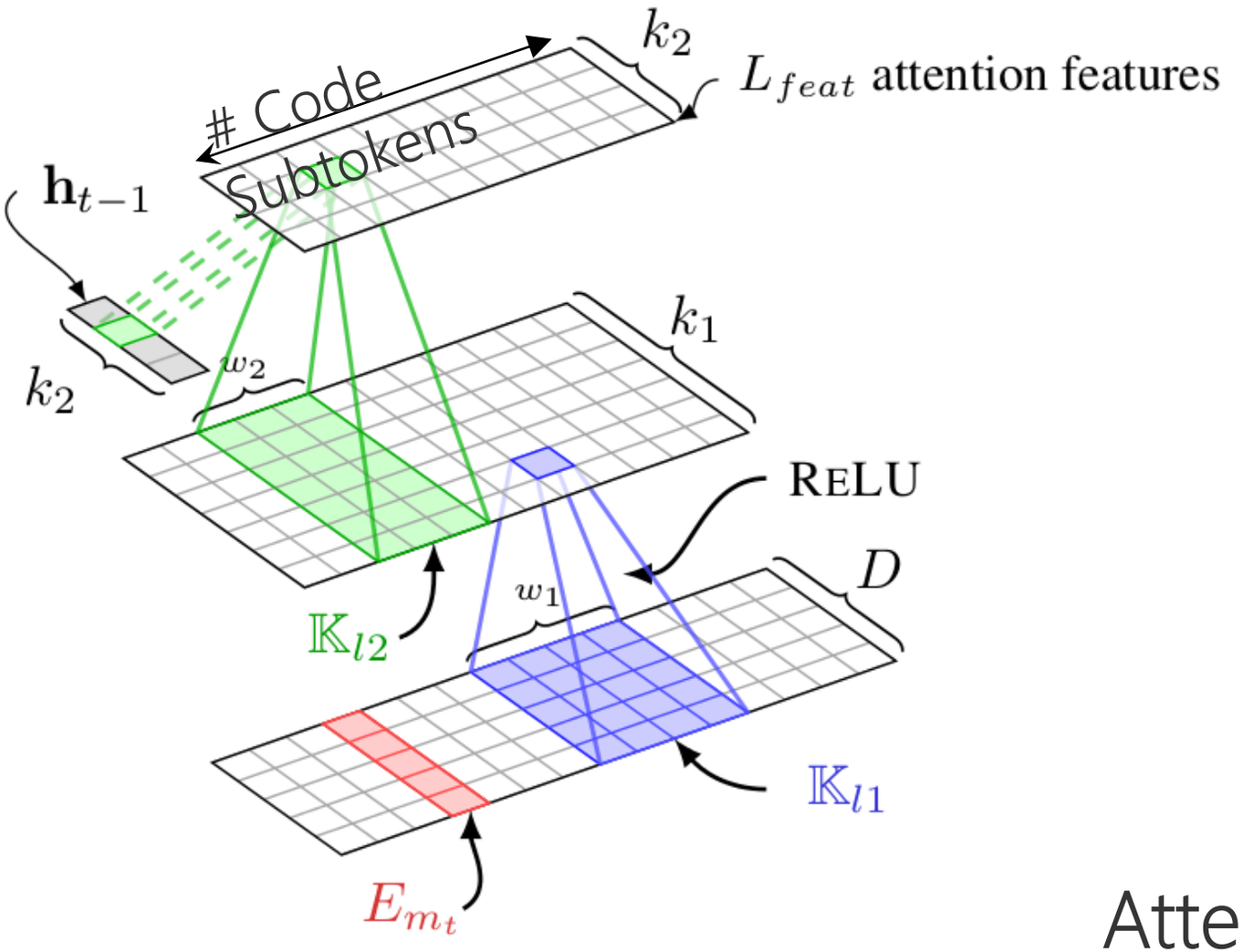
Embedding
Code
Subtokens

```
{
    synchronized(sGlThreadManager) {
      return mRenderMode;
    }
}
```

Extracting Attention
Features

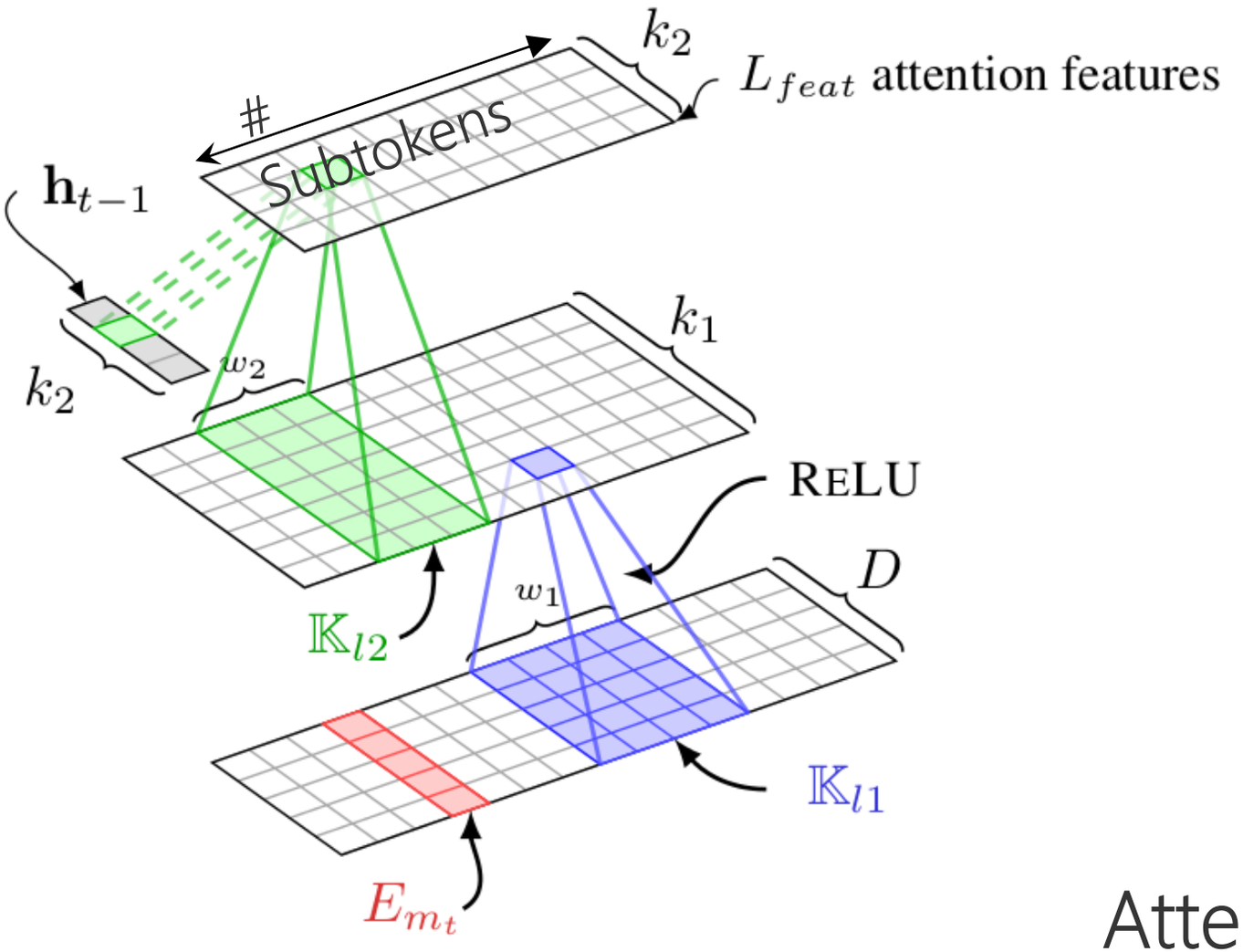


Extracting Attention Features

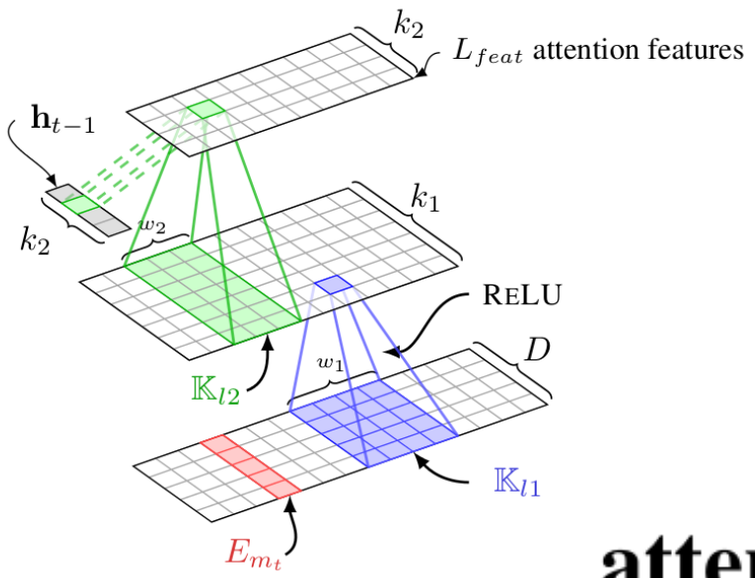


Computing Attention Features

Context-dependent



Computing Attention Features



attention_features (code tokens $\mathbf{c}$, context $\mathbf{h}_{t-1}$)

$C \leftarrow \textsc{LookupAndPad}(\mathbf{c}, E)$

$L_1 \leftarrow \textsc{ReLU}(\textsc{Conv1D}(C, \mathbb{K}_{l1}))$

$L_2 \leftarrow \textsc{Conv1D}(L_1, \mathbb{K}_{l2}) \odot \mathbf{h}_{t-1}$

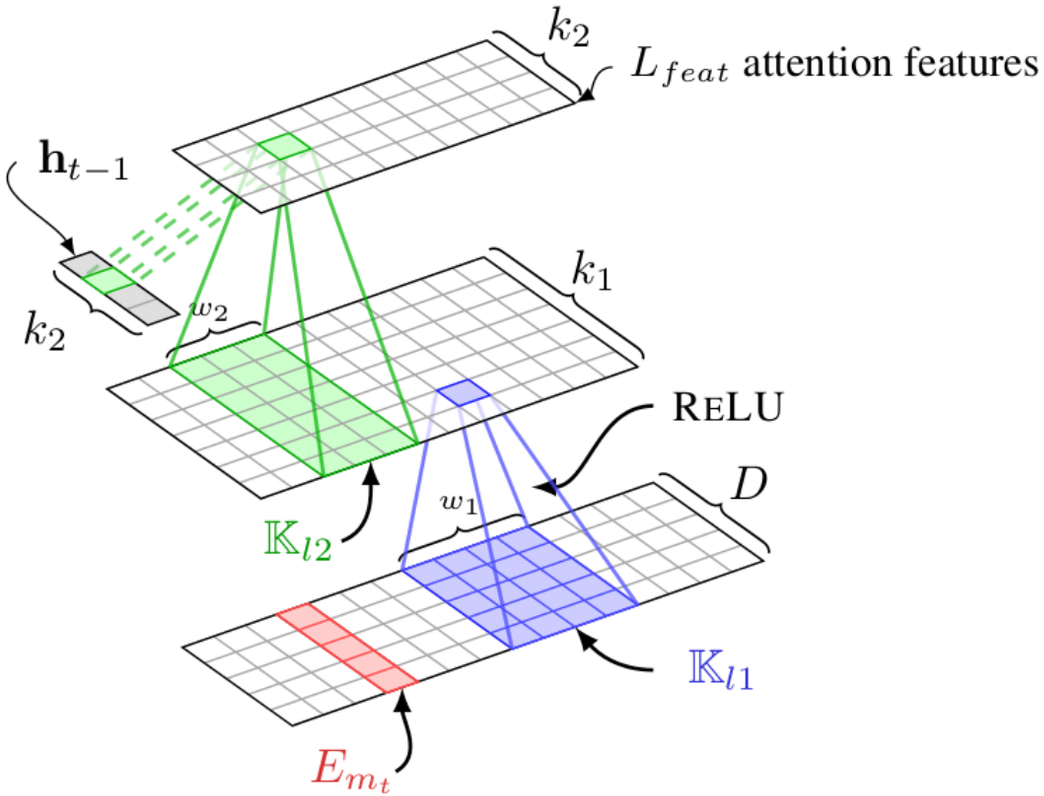
$L_{feat} \leftarrow L_2 / \|L_2\|_2$

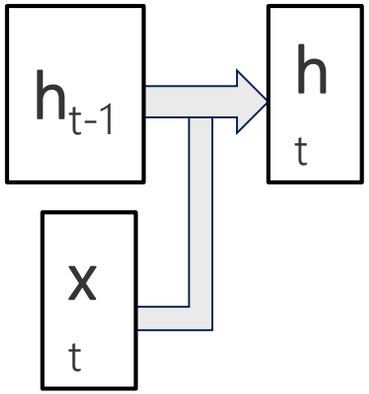
return L_{feat}

Extracting Attention Features

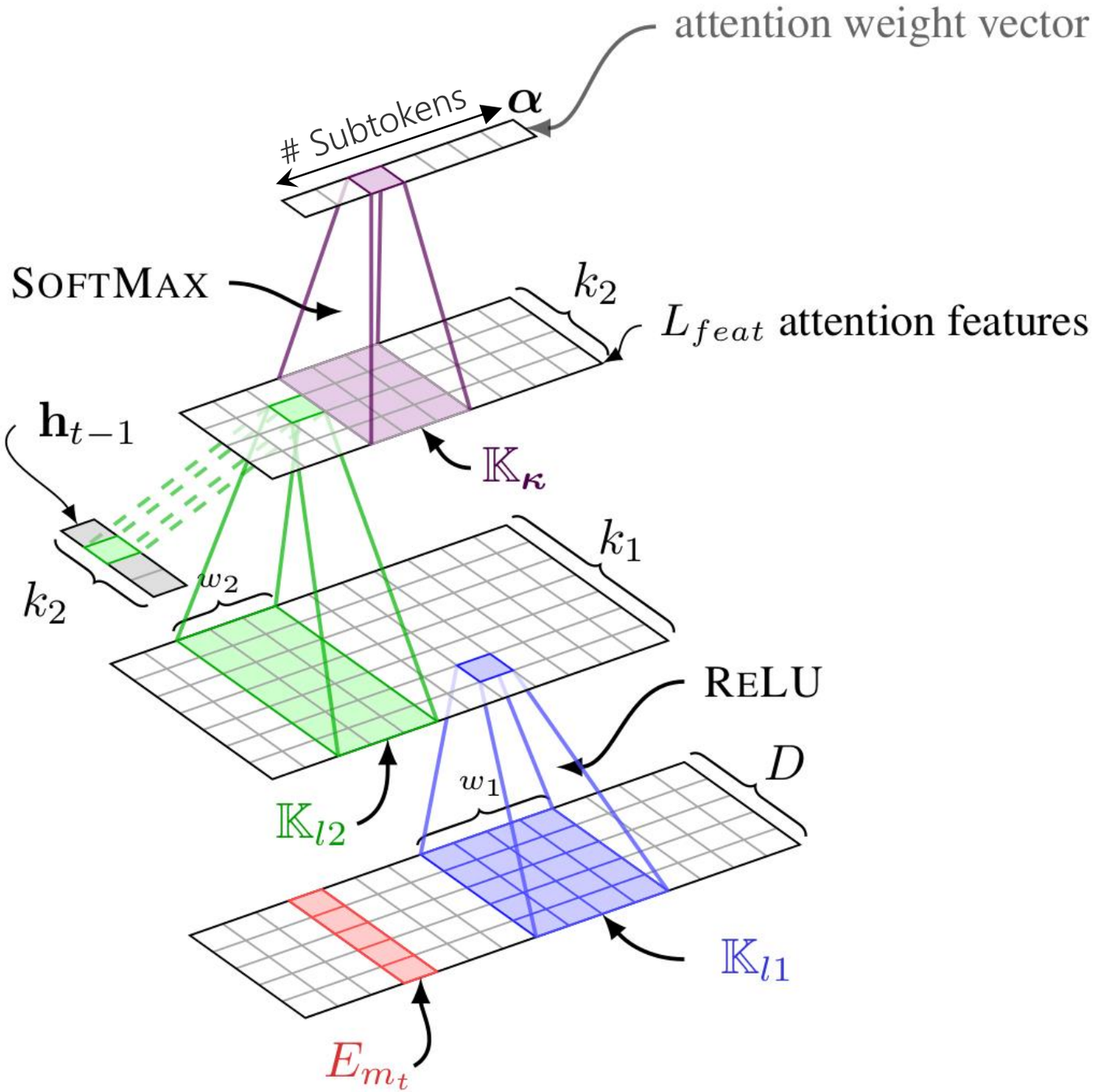
GRU(current input $\mathbf{x}_t$, previous state $\mathbf{h}_{t-1}$)

$$\mathbf{r}_t \leftarrow \sigma(\mathbf{x}_t W_{xr} + \mathbf{h}_{t-1} W_{hr} + \mathbf{b}_r)$$
$$\mathbf{u}_t \leftarrow \sigma(\mathbf{x}_t W_{xu} + \mathbf{h}_{t-1} W_{hu} + \mathbf{b}_u)$$
$$\mathbf{c}_t \leftarrow \tanh(\mathbf{x}_t W_{xc} + r_t \circ (\mathbf{h}_{t-1} W_{hc}) + \mathbf{b}_c)$$
$$\mathbf{h}_t \leftarrow (1 - \mathbf{u}_t) \circ \mathbf{h}_{t-1} + \mathbf{u}_t \circ \mathbf{c}_t$$
return $\mathbf{h}_t$



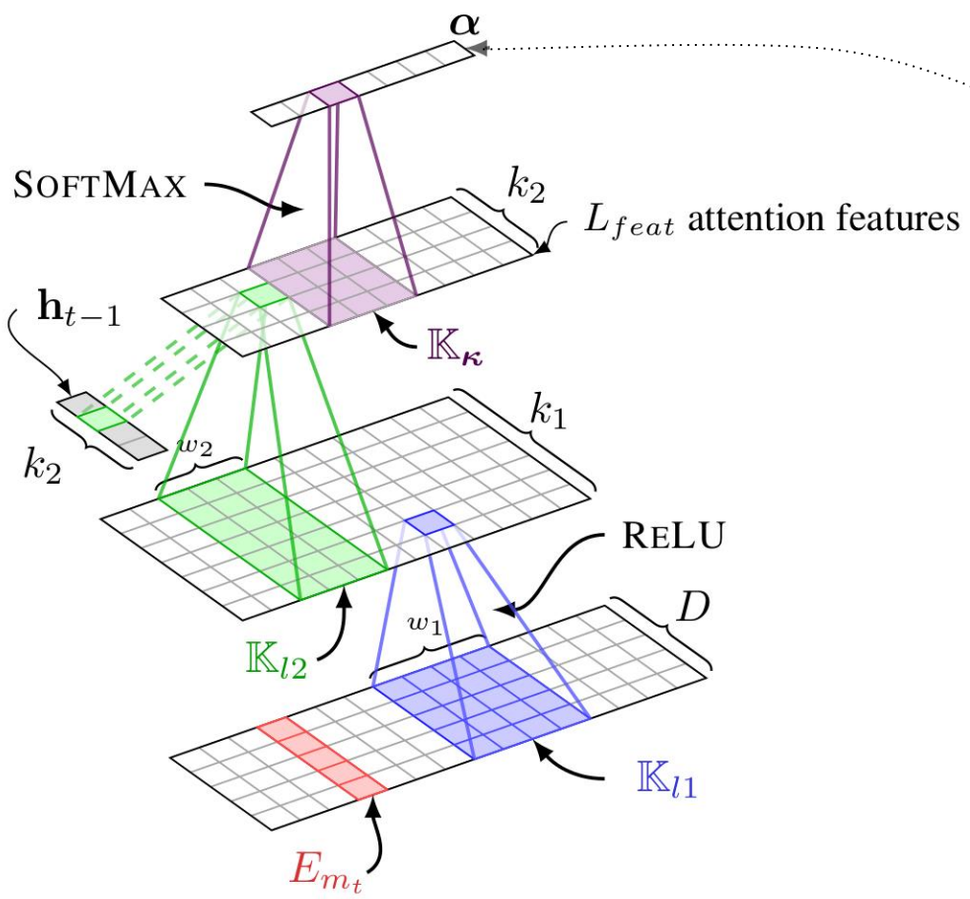


Cho et al., 2014



Computing Attention Weights

Predicting the Next Method Subtoken



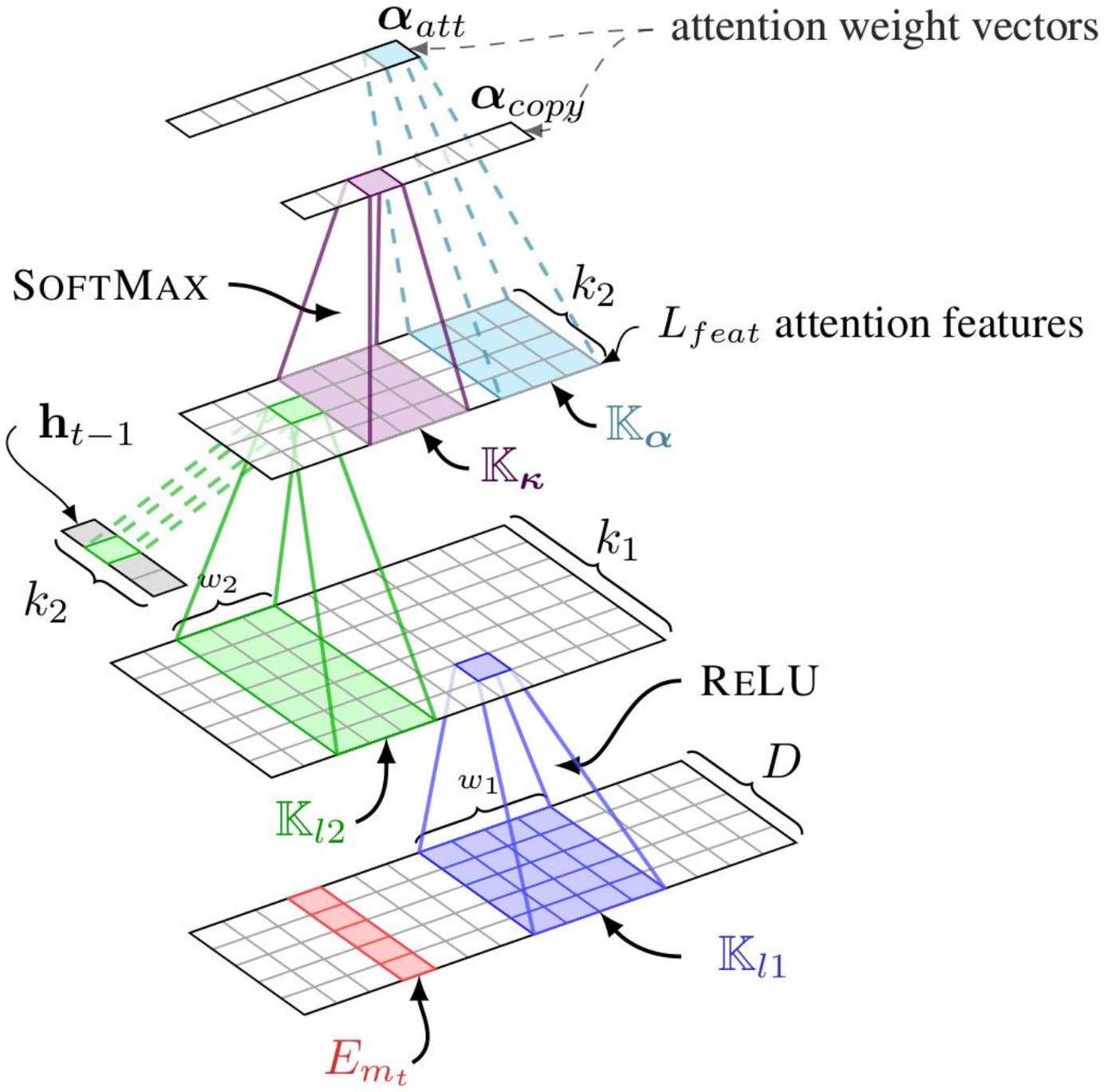
conv_attention (code **c**, previous state $\mathbf{h}_{t-1}$)

$\boldsymbol{\alpha} \leftarrow$ **attention_weights** $(L_{feat}, \mathbb{K}_{att})$

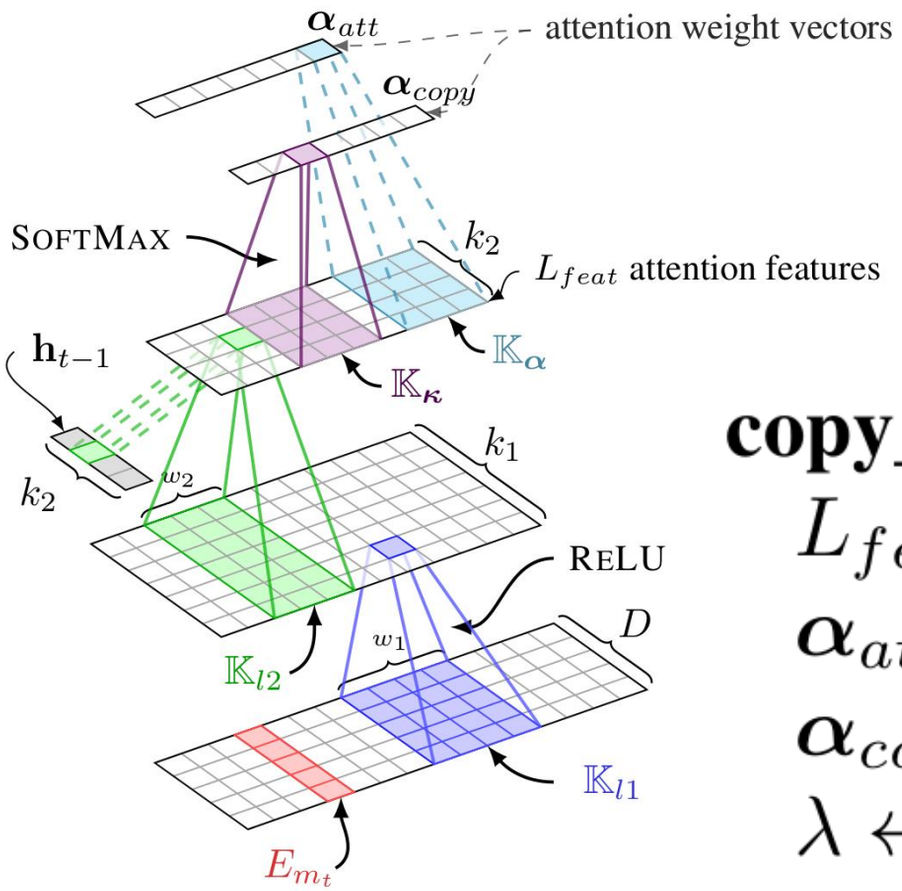
$$\hat{\mathbf{n}} \leftarrow \sum_i \alpha_i E_{c_i}$$

$$\mathbf{n} \leftarrow \textsc{SoftMax}(E\,\hat{\mathbf{n}}^\top + \mathbf{b})$$

return n



Computing Multiple Attention Weights



copy_attention (code $\mathbf{c}$, previous state $\mathbf{h}_{t-1}$)

$L_{feat} \leftarrow$ **attention_features** $(\mathbf{c}, \mathbf{h}_{t-1})$

$\boldsymbol{\alpha}_{att} \leftarrow$ **attention_weights** $(L_{feat}, \mathbb{K}_{att})$

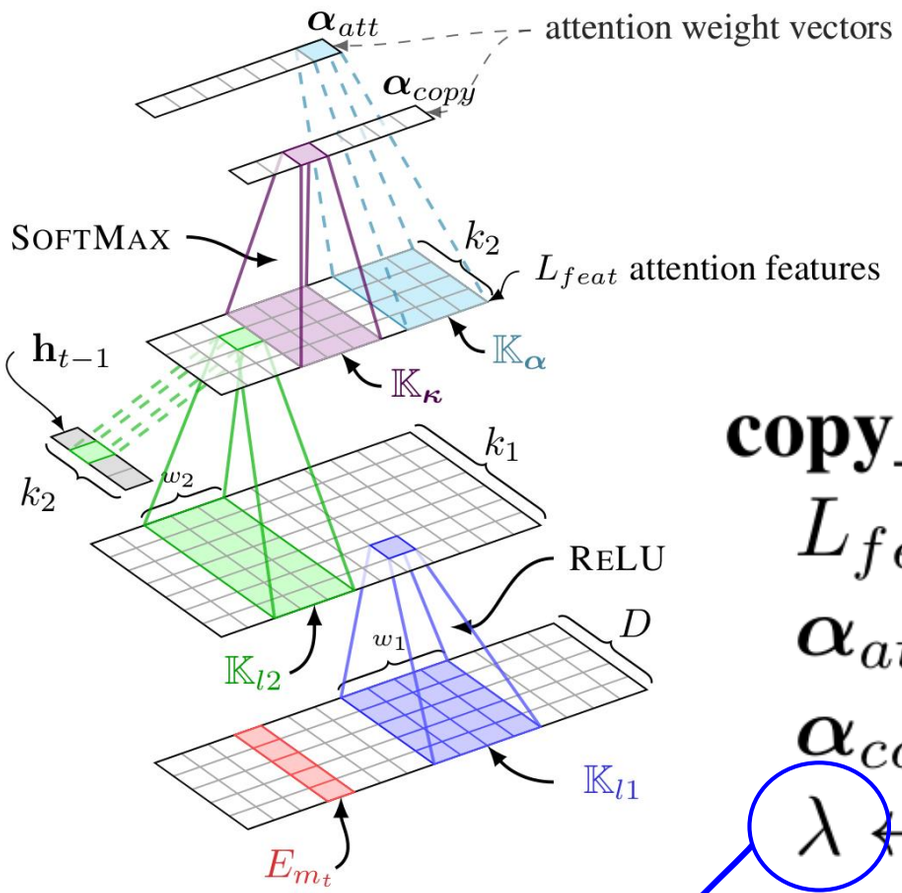
$\boldsymbol{\alpha}_{copy} \leftarrow$ **attention_weights** $(L_{feat}, \mathbb{K}_{copy})$

$\lambda \leftarrow \max(\sigma(\textsc{Conv1D}(L_{feat}, \mathbb{K}_{\lambda})))$

$\hat{\mathbf{n}} \leftarrow \sum_i \alpha_i E_{c_i}$

$\mathbf{n} \leftarrow \textsc{SoftMax}(E\,\hat{\mathbf{n}}^{\top} + \mathbf{b})$

return $\lambda \textsc{Pos2Voc}(\boldsymbol{\kappa}, \mathbf{c}) + (1-\lambda)\textsc{ToMap}(\mathbf{n}, V)$



copy_attention (code $\mathbf{c}$, previous state $\mathbf{h}_{t-1}$)

$L_{feat} \leftarrow$ **attention_features** $(\mathbf{c}, \mathbf{h}_{t-1})$

$\boldsymbol{\alpha}_{att} \leftarrow$ **attention_weights** $(L_{feat}, \mathbb{K}_{att})$

$\boldsymbol{\alpha}_{copy} \leftarrow$ **attention_weights** $(L_{feat}, \mathbb{K}_{copy})$

$\lambda \leftarrow \max(\sigma(\textsc{Conv1D}(L_{feat}, \mathbb{K}_{\lambda})))$

$\hat{\mathbf{n}} \leftarrow \sum_i \alpha_i E_{c_i}$

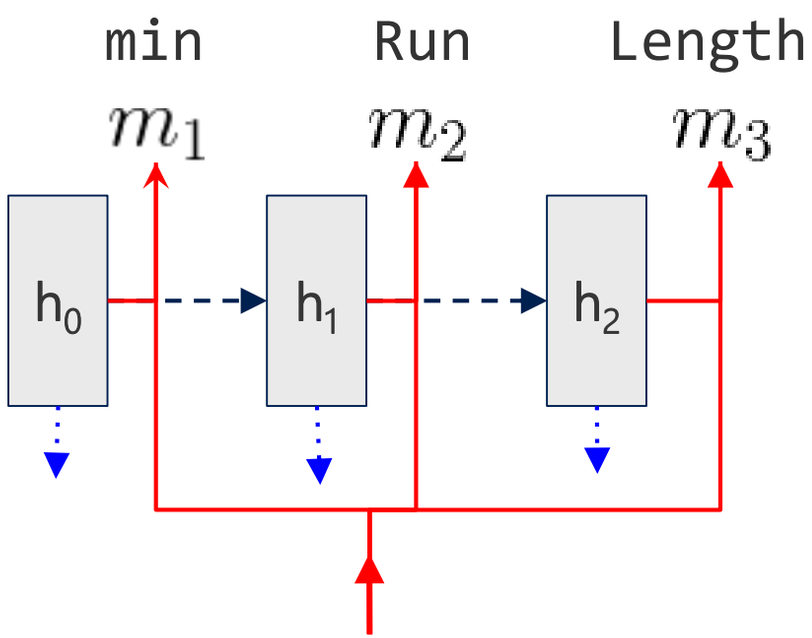
$\mathbf{n} \leftarrow \textsc{SoftMax}(E\,\hat{\mathbf{n}}^{\top} + \mathbf{b})$

return $\lambda \textsc{Pos2Voc}(\boldsymbol{\kappa}, \mathbf{c}) + (1 - \lambda)\textsc{ToMap}(\mathbf{n}, V)$

meta-attention mechanism

Architecture

(Subtoken) Summary



An RNN to predict summary subtokens

$$P(m_i | m_0 \ldots m_{i-1}, \mathbf{code})$$

Code

```
if (DEBUG) assert n >= 0;
int r = 0;
while (n >= MIN_MERGE) {
  r |= (n & 1);
  n >>= 1;
}
return n + r;
```

Context-Dependent Convolutional Attention Features

Project	Forks	Stars	# Train Samples	Description
cassandra	1,275	2,795	11,490	Distributed Database
elasticsearch	5,726	16,923	15,330	REST Search Engine
gradle	1,407	3,581	11,860	Build System
hadoop-common	926	610	25,864	Map-Reduce Framework
hibernate-orm	1,511	1,858	17,950	Object-Relational Mapping
intellij-community	1,355	3,119	236,365	IDE
liferay-portal	1,545	889	70,389	Portal Framework
presto	1,437	4,815	8,584	Distributed SQL query engine
spring-framework	7,559	9,160	23,372	Application Framework
wildfly	1,594	1,620	23,365	Application Server

Dataset

http://groups.inf.ed.ac.uk/cup/codeattention/

	F1 (%)		Exact Match (%)	
	Rank 1	Rank 5	Rank 1	Rank 5
tf-idf	40.0	52.1	**24.3**	29.3
Standard Attention	33.6	45.2	17.4	24.9
No Copy Network	43.6	57.7	20.6	29.8
Full Network	**44.7**	**59.6**	23.5	**33.7**

Bahdanau, Cho, and Bengio. *"Neural machine translation by jointly learning to align and translate"*. 2015

Results — Predicting Method Names

Target Name	set	use	browser	cache
	m_1	m_2	m_3	m_4

	Target		Attention Vectors	λ
m_1	set	$\boldsymbol{\alpha}_{att} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.012
		$\boldsymbol{\alpha}_{copy} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_2	use	$\boldsymbol{\alpha}_{att} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.974
		$\boldsymbol{\alpha}_{copy} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_3	browser	$\boldsymbol{\alpha}_{att} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.969
		$\boldsymbol{\alpha}_{copy} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_4	cache	$\boldsymbol{\alpha}_{att} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.583
		$\boldsymbol{\alpha}_{copy} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	
m_5	END	$\boldsymbol{\alpha}_{att} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	0.066
		$\boldsymbol{\alpha}_{copy} =$	<s> { this . use Browser Cache = use Browser Cache ; } </s>	

Attention Visualization

Data and Visualizations:
http://groups.inf.ed.ac.uk/cup/codeattention/

Target Name

`is`	`bullet`
m_1	m_2

Target		Attention Vectors		λ
m_1	`is`	$\alpha =$	`<s> { return ( m Flags & e Bullet Flag ) == e Bullet Flag ; } </s>`	0.012
		$\kappa =$	`<s> { return ( m Flags & e Bullet Flag ) == e Bullet Flag ; } </s>`	
m_2	`bullet`	$\alpha =$	`<s> { return ( m Flags & e Bullet Flag ) == e Bullet Flag ; } </s>`	0.436
		$\kappa =$	`<s> { return ( m Flags & e Bullet Flag ) == e Bullet Flag ; } </s>`	
m_3	END	$\alpha =$	`<s> { return ( m Flags & e Bullet Flag ) == e Bullet Flag ; } </s>`	0.174
		$\kappa =$	`<s> { return ( m Flags & e Bullet Flag ) == e Bullet Flag ; } </s>`	

Attention Visualization

Data and Visualizations:
http://groups.inf.ed.ac.uk/cup/codeattention/

```
void reverseRange(Object[] a, int lo, int hi)
```

```
hi--;
while (lo < hi) {
  Object t = a[lo];
  a[lo++] = a[hi];
  a[hi--] = t;
}
```

Suggestions

- `reverse, range` (22%)
- `reverse` (13%)
- `reverse,lo` (4%)
- `reverse,hi` (3%)

int createProgram()

```
GL20 gl = Gdx.gl20;
int program = gl.glCreateProgram();
return program != 0 ? program : -1;
```

Suggestions

- create (18%)
- init (8%)
- render (5%)
- initiate (5%)

```
float getAspectRatio()
```

```
return (height == 0) ?
    Float.NaN : width / height;
```

Predictions

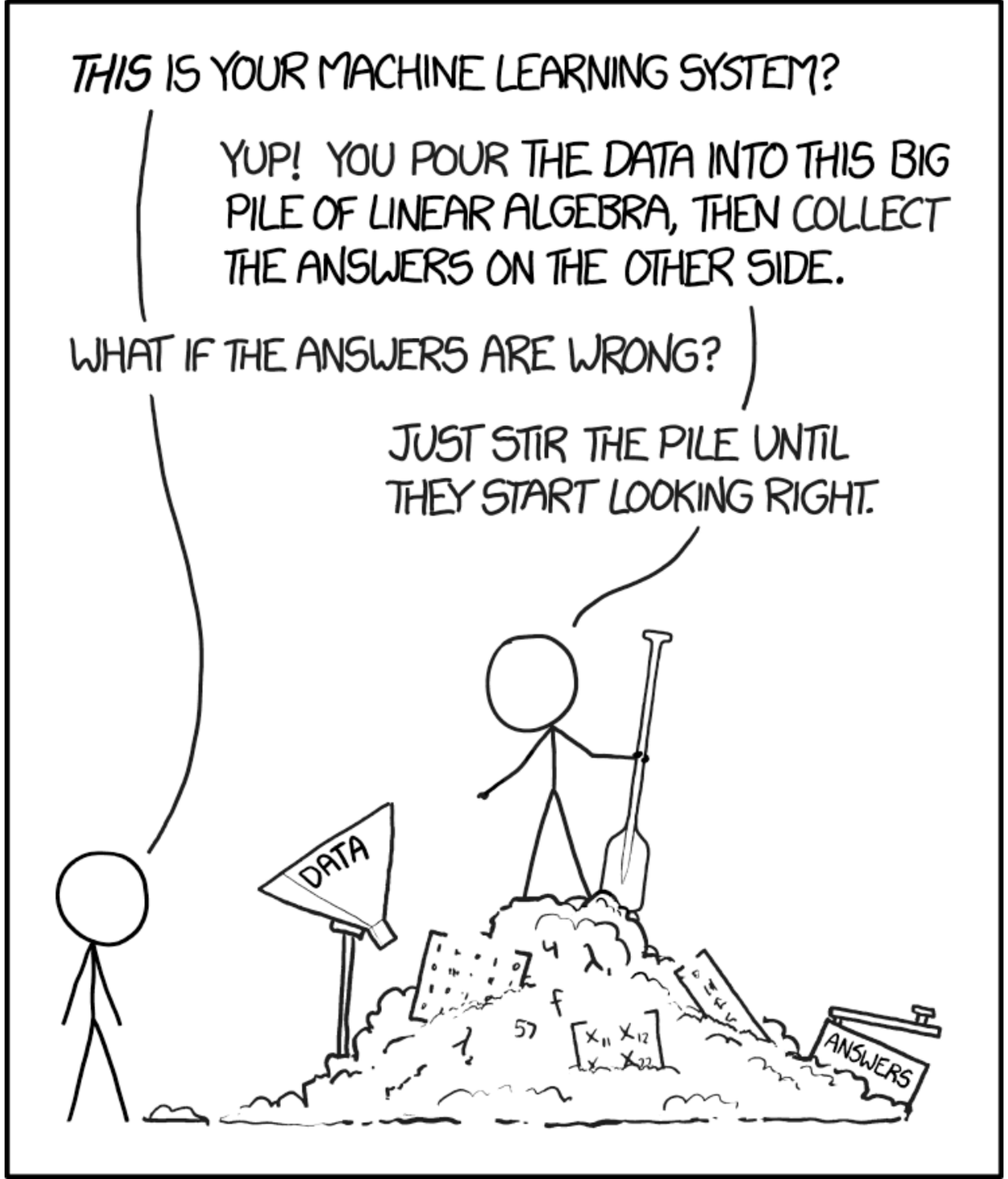
- `get, UNK` (9%)
- `get, height` (8.7%)
- `get, width` (6.5%)
- `get` (5.7%)
- `get,size` (4.2%)

```
boolean shouldRender()
```

```
try {
  return renderRequested||isContinuous;
} finally {
  renderRequested = false;
}
```

Predictions

- `is, render` (27%)
- `is,continuous` (11%)
- `is,requested` (8%)
- `render,continuous` (7%)


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

Model Capacity (*what can the model learn?*)

- Overtrain on a small dataset
- Synthetic data

Optimization Issues (*can we make the model learn?*)

- Look at training curves
- Monitor gradient update ratios
- Hand-pick parameters for synthetic data

Other model "bugs" (*is the model doing what I want it to do?*)

- Generate samples from your model (if you can)
- Visualize learned representations (*e.g.* embeddings, nearest neighbors)
- Error analysis (examples where the model is failing, most "confident" errors)
- Simplify the problem/model
- Increase capacity, sweep hyperparameters (*e.g.* increase size of **h** in LSTM*)*

Practical(?) Tips on Debugging Machine Learning Models

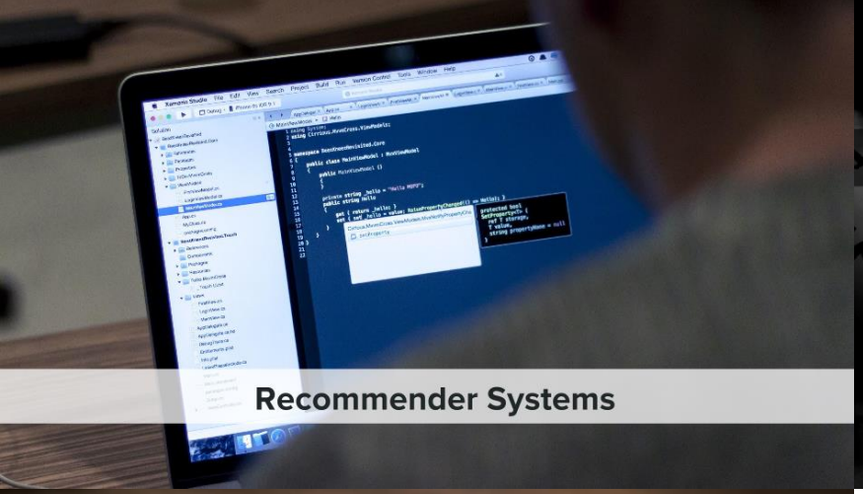
Multiplicative LSTM for sequence modelling.
Ben Krause, Iain Murray, Steve Renals, and Liang Lu.

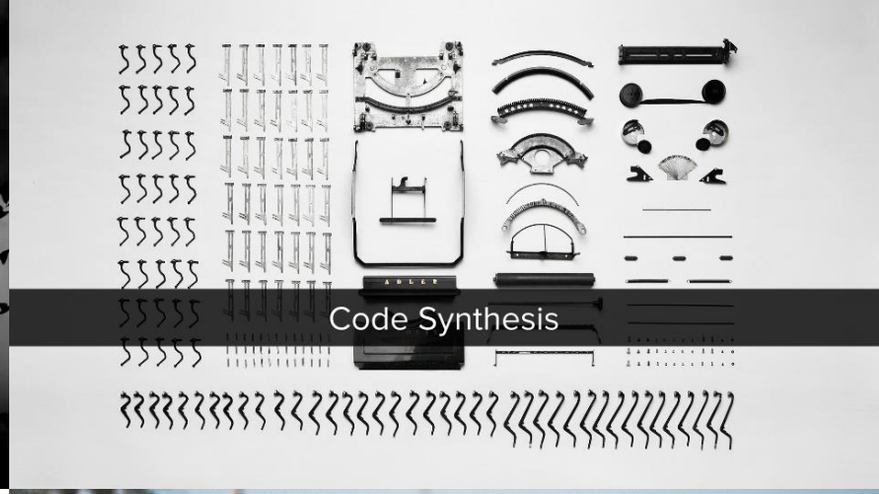
Learning Python Code Suggestion with a Sparse Pointer Network.
Bhoopchand, Avishkar, *et al*.

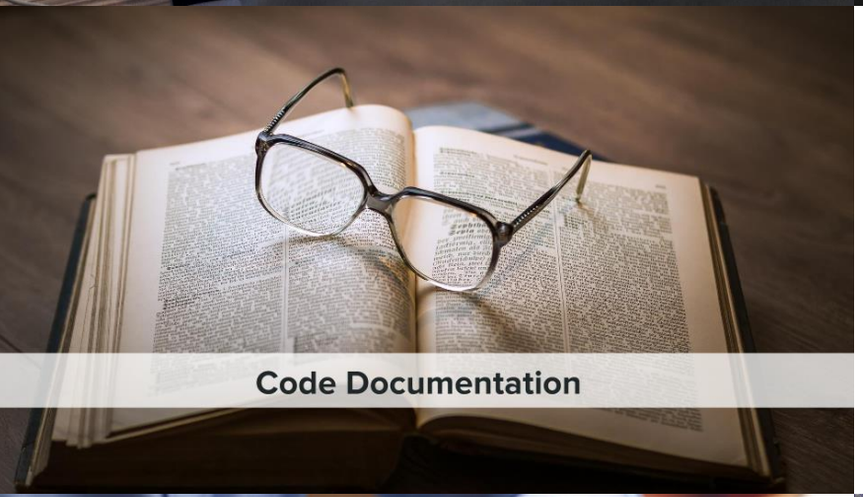
Sub-token Language Models

Attention over symbol table

Some More Code Completion Ideas…







Machine Learning Models