



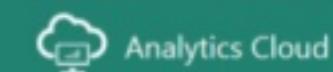
**SPARK
SUMMIT
2017**

Neuro-Symbolic Sentiment Analysis

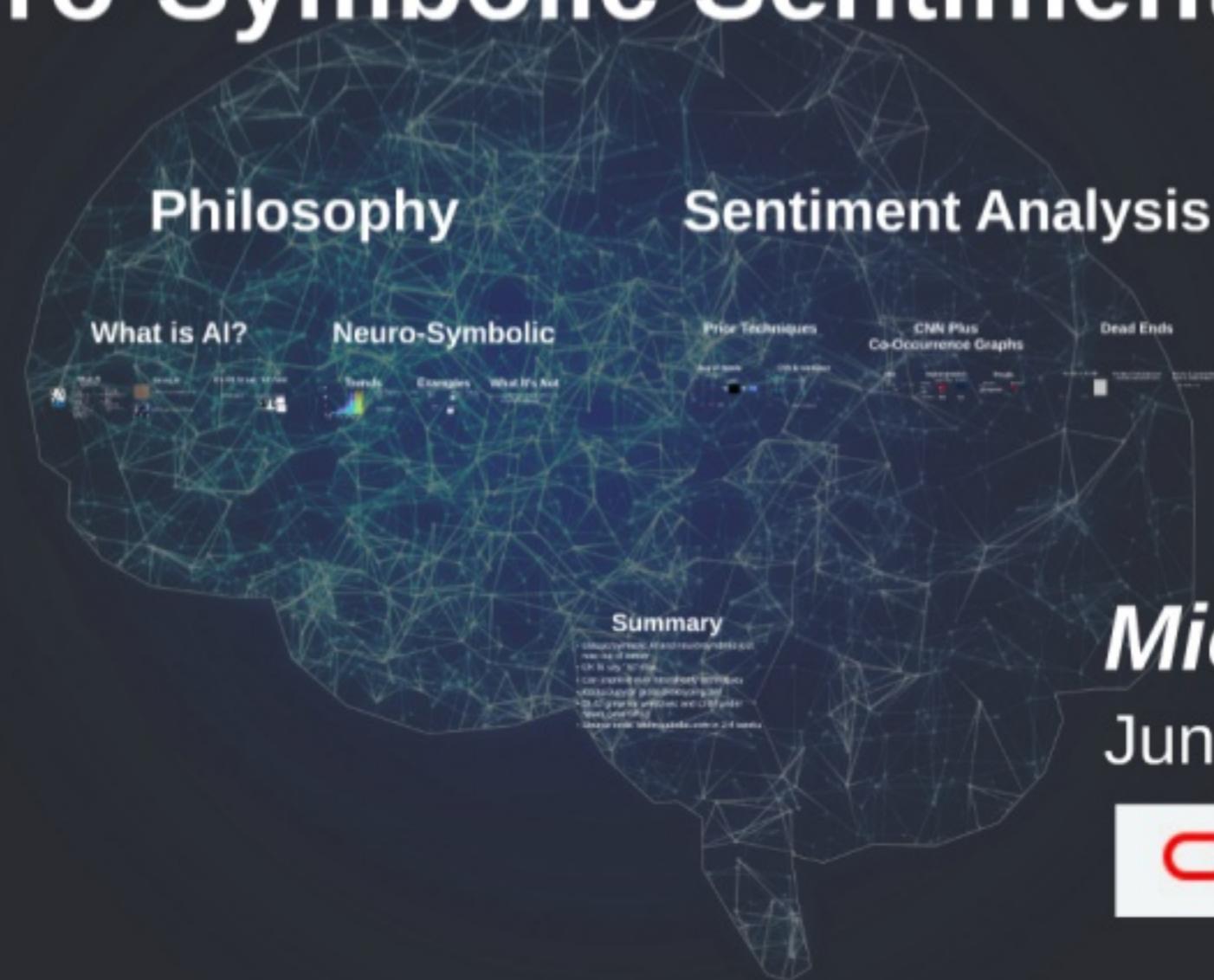


Michael Malak

June 7, 2017



Neuro-Symbolic Sentiment Analysis



Michael Malak

June 7, 2017

ORACLE® Cloud



Big Data Preparation



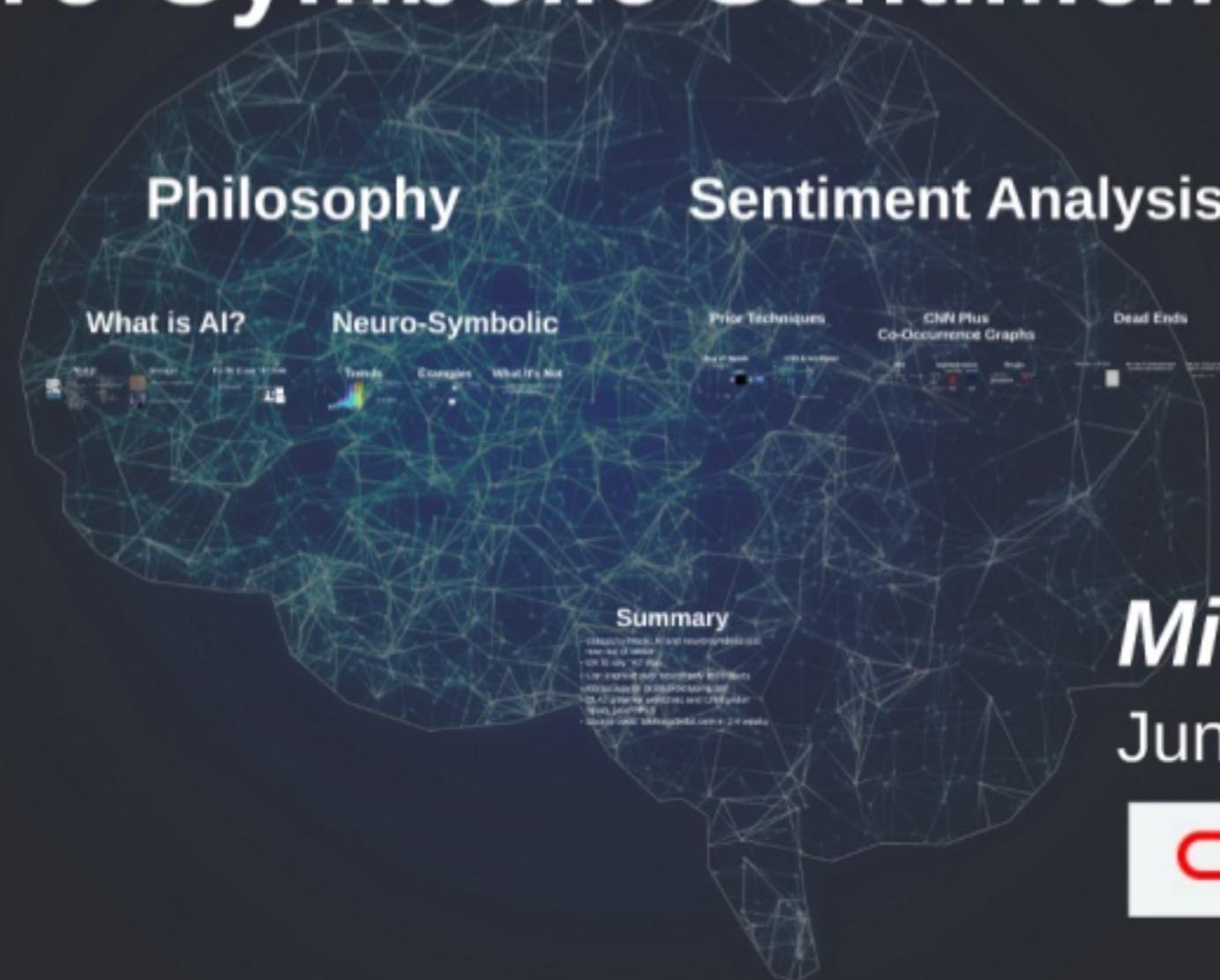
Analytics Cloud

Disclaimer

The information in this presentation is intended for informational purposes only, and may not be incorporated into any contract. This is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Copyright © 2017 Oracle and/or its affiliates. All rights reserved.
Patent Pending.

Neuro-Symbolic Sentiment Analysis



Michael Malak

June 7, 2017

ORACLE® Cloud



Big Data Preparation



Analytics Cloud

Philosophy

What is AI?

Neuro-Symbolic

Weak AI
Classic AI



Strong AI



Artificial General Intelligence (AGI)

Artificial Super Intelligence (ASI)

It's OK to say "AI" now

AI Permafrost

Silicon Valley AI Index

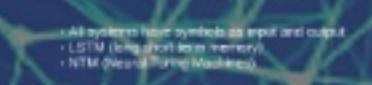
Trends



Examples



What It's Not



- AI systems have symbols as input and output
- LSTM (long short-term memory)
- NTM (Neural Turing Machine)

What is AI?

Weak AI *Classic AI*



Part I Artificial Intelligence
1 Introduction
2 Intelligent Agents

Part II Problem Solving
3 Solving Problems by Searching
4 Beyond Classical Search
5 Adaptive Search
6 Constraint Satisfaction Problems

Part III Knowledge and Reasoning
7 Logical Agents
8 First-Order Logic
9 Inference in First-Order Logic
10 Classical Planning
11 Planning and Acting in the Real World
12 Knowledge Representation

Part IV Uncertain Knowledge and Reasoning
13 Quantifying Uncertainty
14 Probabilistic Reasoning
15 Probabilistic Reasoning over Time
16 Making Single Decisions
17 Making Complex Decisions

Part V Learning
18 Learning from Examples
19 Knowledge in Learning
20 Learning Probabilistic Models
21 Reinforcement Learning

Part VI Communicating, Perceiving, and Acting
22 Natural Language Processing
23 Natural Language for Communication
24 Perception
25 Robotics

Part VII Conclusions
26 Philosophical Foundations
27 AI: The Present and Future
A Mathematical Background
B Notes on Languages and Algorithms
Index



Strong AI

Artificial General Intelligence (AGI)

Artificial Super Intelligence (ASI)

It's OK to say "AI" now

AI Permafrost

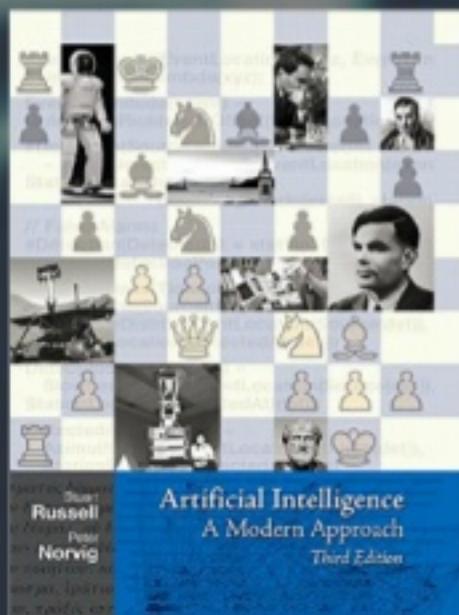


Silicon Valley AI Depts



Weak AI

Classic AI



Part I Artificial Intelligence

- 1 Introduction
- 2 Intelligent Agents

Part II Problem Solving

- 3 Solving Problems by Searching
- 4 Beyond Classical Search
- 5 Adversarial Search
- 6 Constraint Satisfaction Problems

Part III Knowledge and Reasoning

- 7 Logical Agents
- 8 First-Order Logic
- 9 Inference in First-Order Logic
- 10 Classical Planning
- 11 Planning and Acting in the Real World
- 12 Knowledge Representation

Part IV Uncertain Knowledge and Reasoning

- 13 Quantifying Uncertainty
- 14 Probabilistic Reasoning
- 15 Probabilistic Reasoning over Time
- 16 Making Simple Decisions
- 17 Making Complex Decisions

Part V Learning

- 18 Learning from Examples
- 19 Knowledge in Learning
- 20 Learning Probabilistic Models
- 21 Reinforcement Learning

Part VI Communicating, Perceiving, and Acting

- 22 Natural Language Processing
- 23 Natural Language for Communication
- 24 Perception
- 25 Robotics

Part VII Conclusions

- 26 Philosophical Foundations
- 27 AI: The Present and Future
- A Mathematical Background
- B Notes on Languages and Algorithms
- Bibliography
- Index

Strong AI



Artificial General Intelligence (AGI)



Artificial Super Intelligence (ASI)

It's OK to say "AI" now

AI Permafrost



Silicon Valley AI Depts



AI Permafrost



Silicon Valley AI Depts



Microsoft expands artificial intelligence (AI) efforts with creation of new Microsoft AI and Research Group

Posted September 26, 2016 by Microsoft News Center

Computer vision luminary Harry Shum to lead more than 5,000 people worldwide

REDMOND, Washington — Sept. 26, 2016 — Microsoft Corp. announced on Thursday it has formed the Microsoft AI and Research Group, bringing together Microsoft's world-class research organization with more than 5,000 computer scientists and engineers focused on the company's AI product efforts. The new group will be led by computer vision luminary Harry Shum, a 20-year Microsoft veteran whose career has spanned leadership roles across Microsoft Research and Bing engineering.

Microsoft is dedicated to democratizing AI for every person and organization, making it more accessible and valuable to everyone and ultimately enabling new ways to solve some of society's toughest challenges. Today's announcement builds on the company's deep focus on AI and will accelerate the delivery of new capabilities to customers across agents, apps, services and infrastructure.

In addition to Shum's existing leadership team, several of the company's engineering leaders and teams will join the newly formed group including Information Platform, Cortana and Bing, and Ambient Computing and Robotics teams led by David Ku, Derrick Connell and Vijay Balaji, respectively. All combined, the Microsoft AI and Research Group will encompass AI product engineering, basic and applied research labs, and New Experiences and Technologies (NET).



Research at Facebook

Home Research Academic Programs Publications Blog People

Facebook AI Research (FAIR)

Research Downloads



Facebook to Accelerate Global AI Research with New GPU Program Recipients

By Michael Sussman | September 26, 2016

The Facebook AI Research (FAIR) program is expanding its global reach, announcing new GPU recipients in India, China, and Europe. The program, which was first launched in 2014, aims to accelerate AI research by providing access to high-end computing resources. The new recipients include institutions such as the University of Edinburgh, University of Cambridge, University of Oxford, and Tsinghua University. The program has already had success in previous years, with notable achievements in areas such as image recognition and natural language processing.

What is AI?

Weak AI *Classic AI*



Part I Artificial Intelligence
1 Introduction
2 Intelligent Agents

Part II Problem Solving
3 Solving Problems by Searching
4 Beyond Classical Search
5 Adaptive Search
6 Constraint Satisfaction Problems

Part III Knowledge and Reasoning
7 Logical Agents
8 First-Order Logic
9 Inference in First-Order Logic
10 Classical Planning
11 Planning and Acting in the Real World
12 Knowledge Representation

Part IV Uncertain Knowledge and Reasoning
13 Quantifying Uncertainty
14 Probabilistic Reasoning
15 Probabilistic Reasoning over Time
16 Making Single Decisions
17 Making Complex Decisions

Part V Learning
18 Learning from Examples
19 Knowledge in Learning
20 Learning Probabilistic Models
21 Reinforcement Learning

Part VI Communicating, Perceiving, and Acting
22 Natural Language Processing
23 Natural Language for Communication
24 Perception
25 Robotics

Part VII Conclusions
26 Philosophical Foundations
27 AI: The Present and Future
28 A Mathematical Background
29 Notes on Languages and Algorithms
30 Bibliography
Index



Strong AI

Artificial General Intelligence (AGI)

Artificial Super Intelligence (ASI)

It's OK to say "AI" now

AI Permafrost

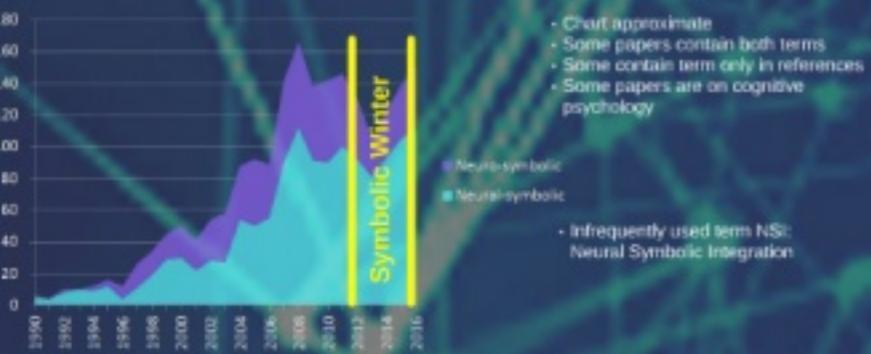


Silicon Valley AI Depts



Neuro-Symbolic

Trends



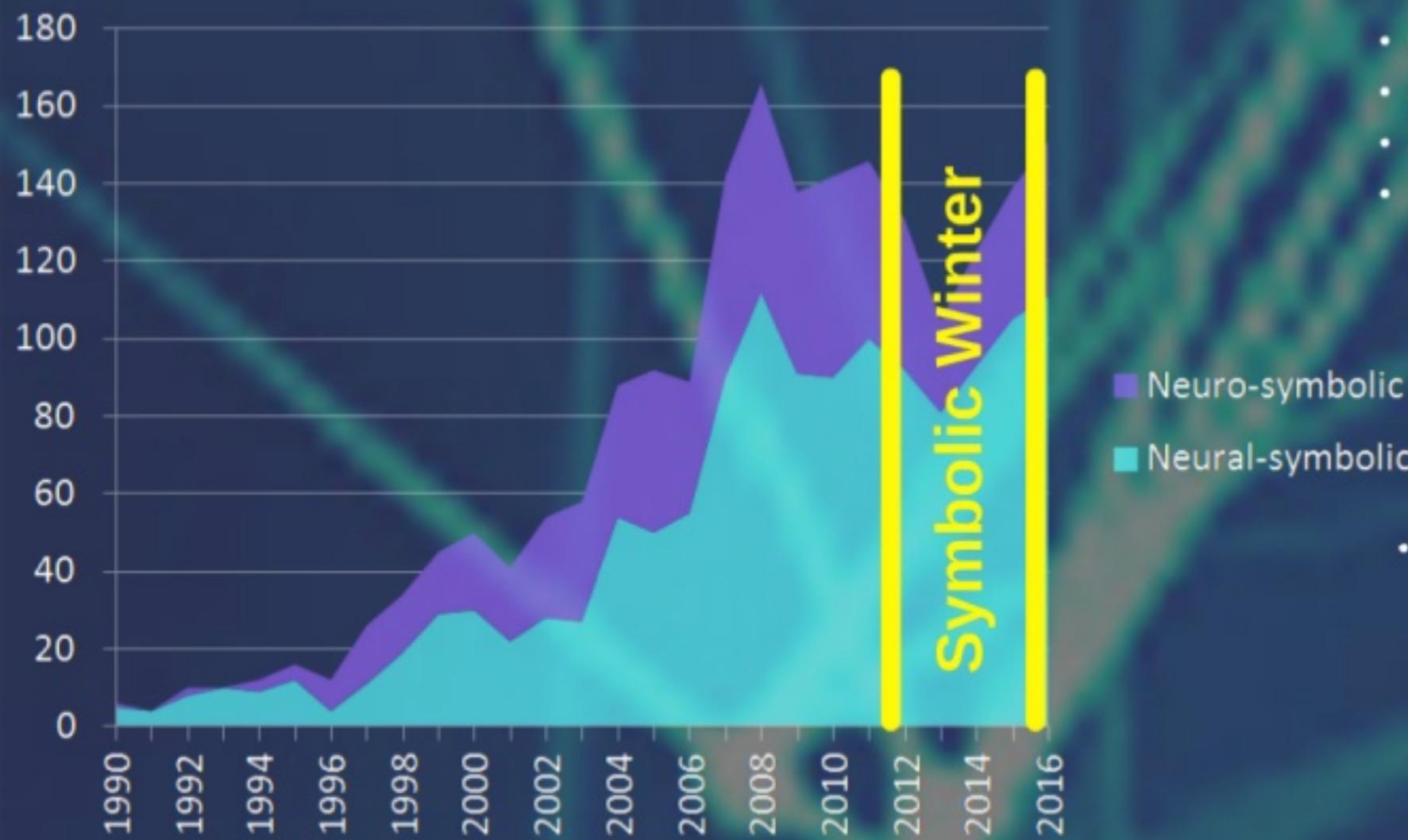
Examples



What It's Not

- All systems have symbols as input and output
- LSTM (long short-term memory)
- NTM (Neural Turing Machines)

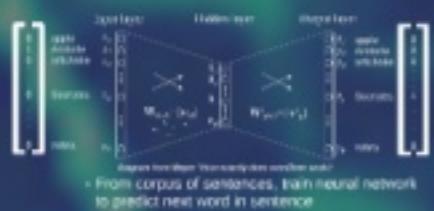
Trends



- Chart approximate
- Some papers contain both terms
- Some contain term only in references
- Some papers are on cognitive psychology
- Infrequently used term NSI: Neural Symbolic Integration

Examples

word2vec



Towards AGI



Enhance neural with symbolic



Symbolic Injection & Extraction



Enhance symbolic with neural



Pipelines



Understanding

Explaining the "black box" of a neural network

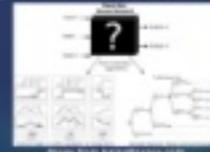


Image from survivingmachine.com
Transfer Learning
One Shot Learning

word2vec

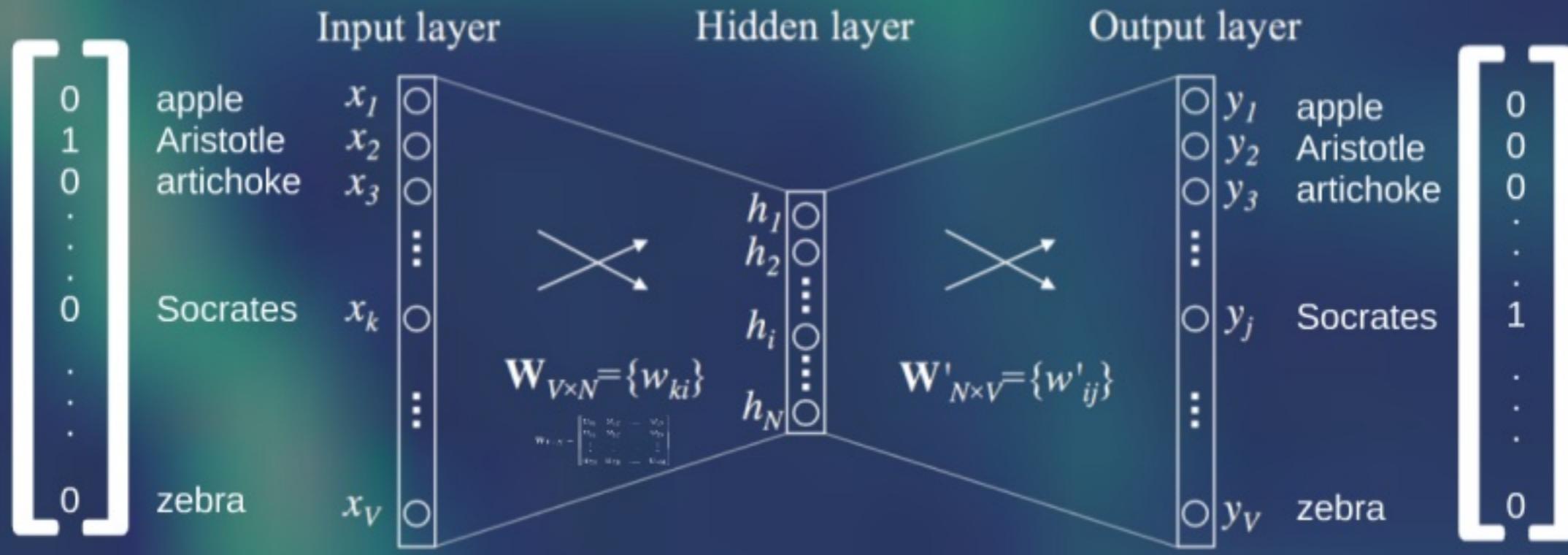


Diagram from Meyer "How exactly does word2vec work?"

- From corpus of sentences, train neural network to predict next word in sentence

$\mathbf{W}_{V \times N} = \{w_{ki}\}$

$$\mathbf{W}_{V \times N} = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1N} \\ w_{21} & w_{22} & \dots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{V1} & w_{V2} & \dots & w_{VN} \end{bmatrix}$$

h_1
 h_2
 h_i
 h_N

\mathbf{W}

word2vec

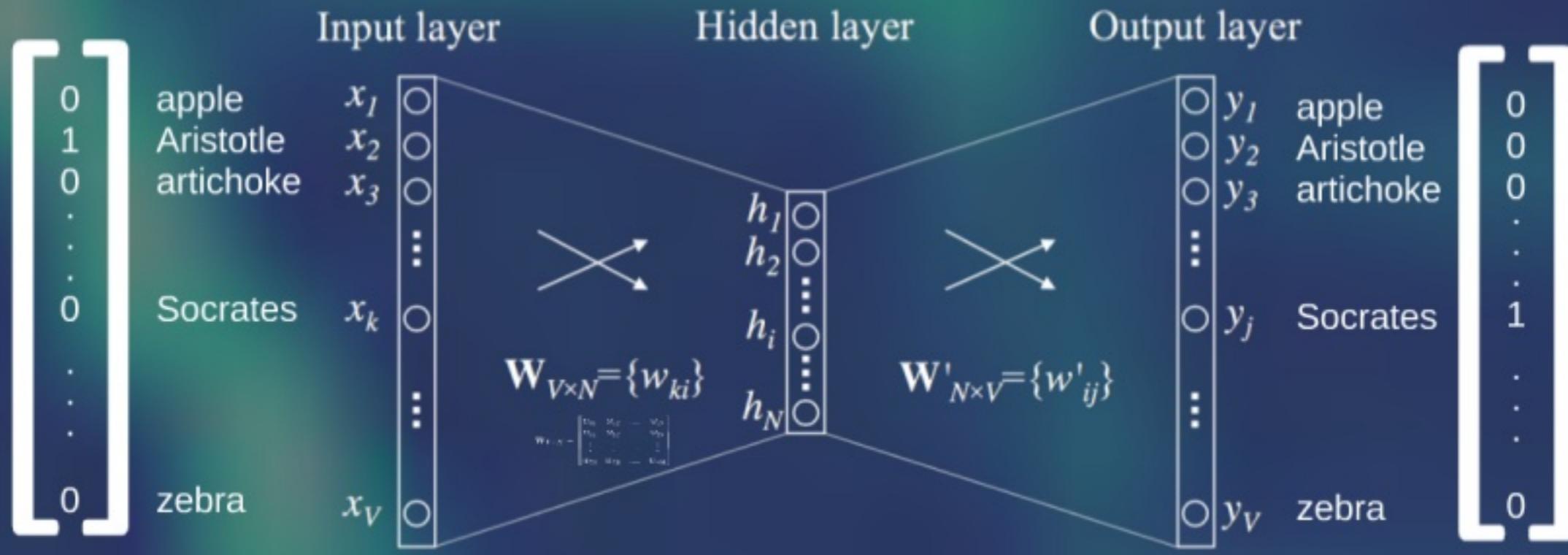
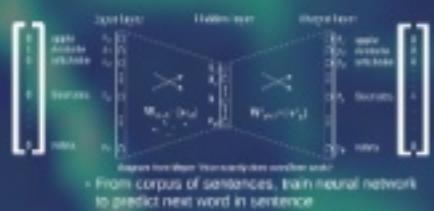


Diagram from Meyer "How exactly does word2vec work?"

- From corpus of sentences, train neural network to predict next word in sentence

Examples

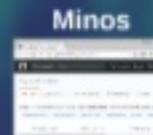
word2vec



Towards AGI



Enhance neural with symbolic



Enhance symbolic with neural



Pipelines



Understanding



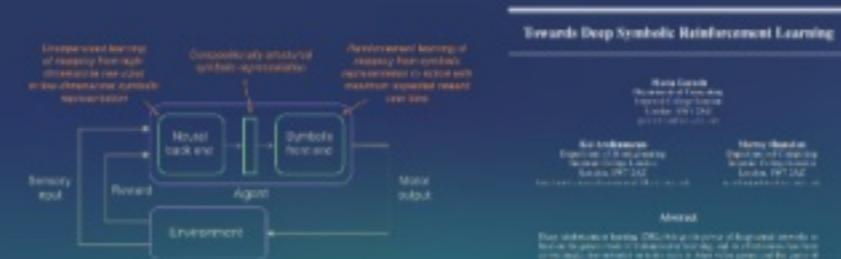
- Explaining the "black box" of a neural network
- Transfer Learning
- One Shot Learning

Towards AGI

OpenCog



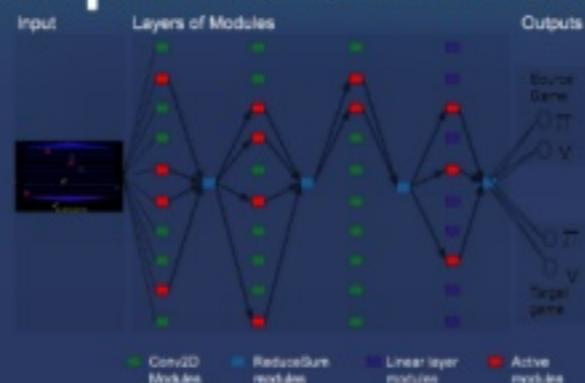
Deep Symbolic Reinforcement Learning



LML: Lifelong Machine Learning



DeepMind's PathNet



OpenCog

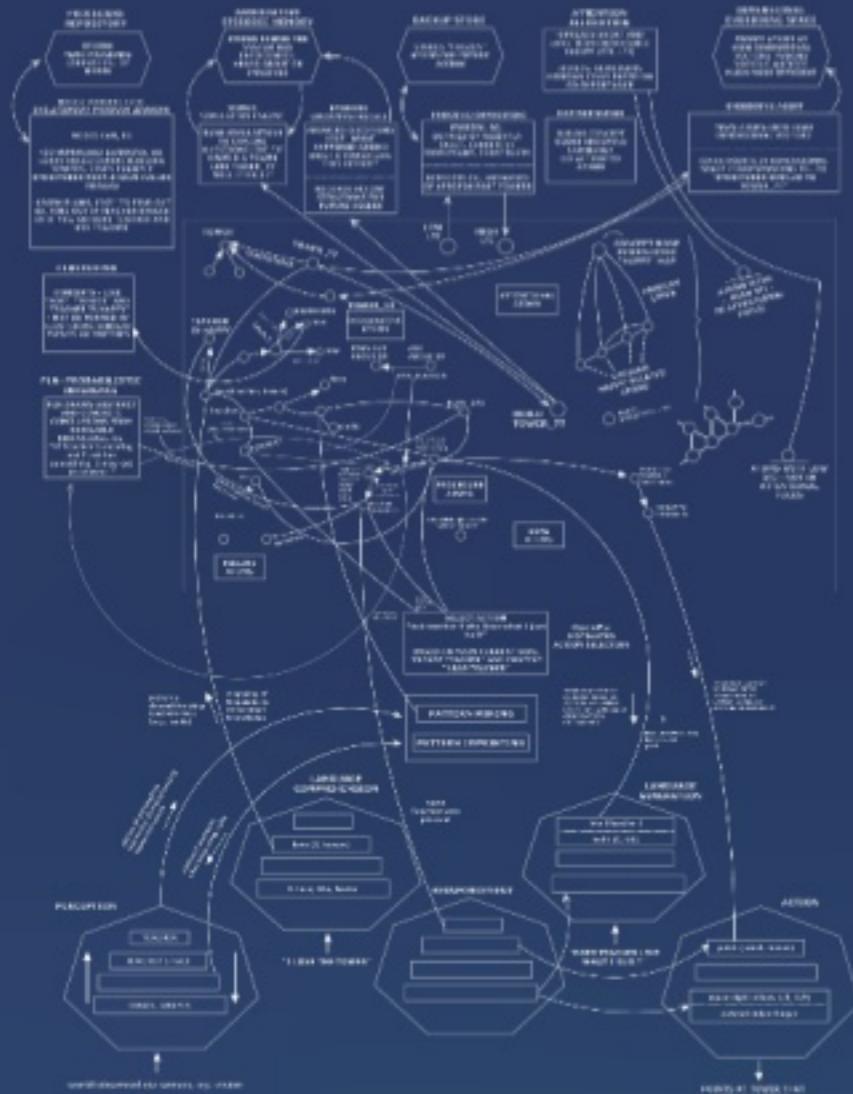


Diagram from goertzel.org

OpenCog

GitHub, Inc. [US] | <https://github.com/opencog>

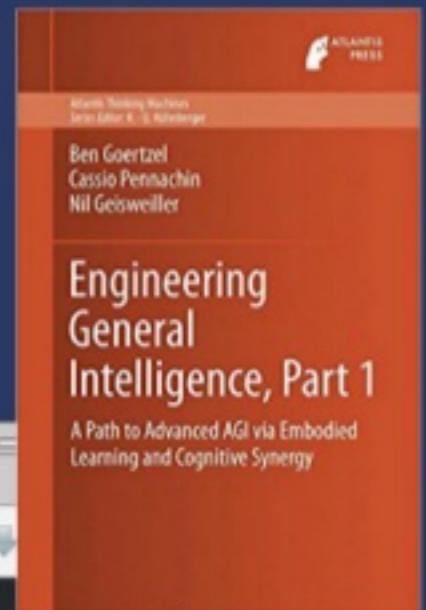
This organization Search Pull requests Issues Marketplace

OpenCog

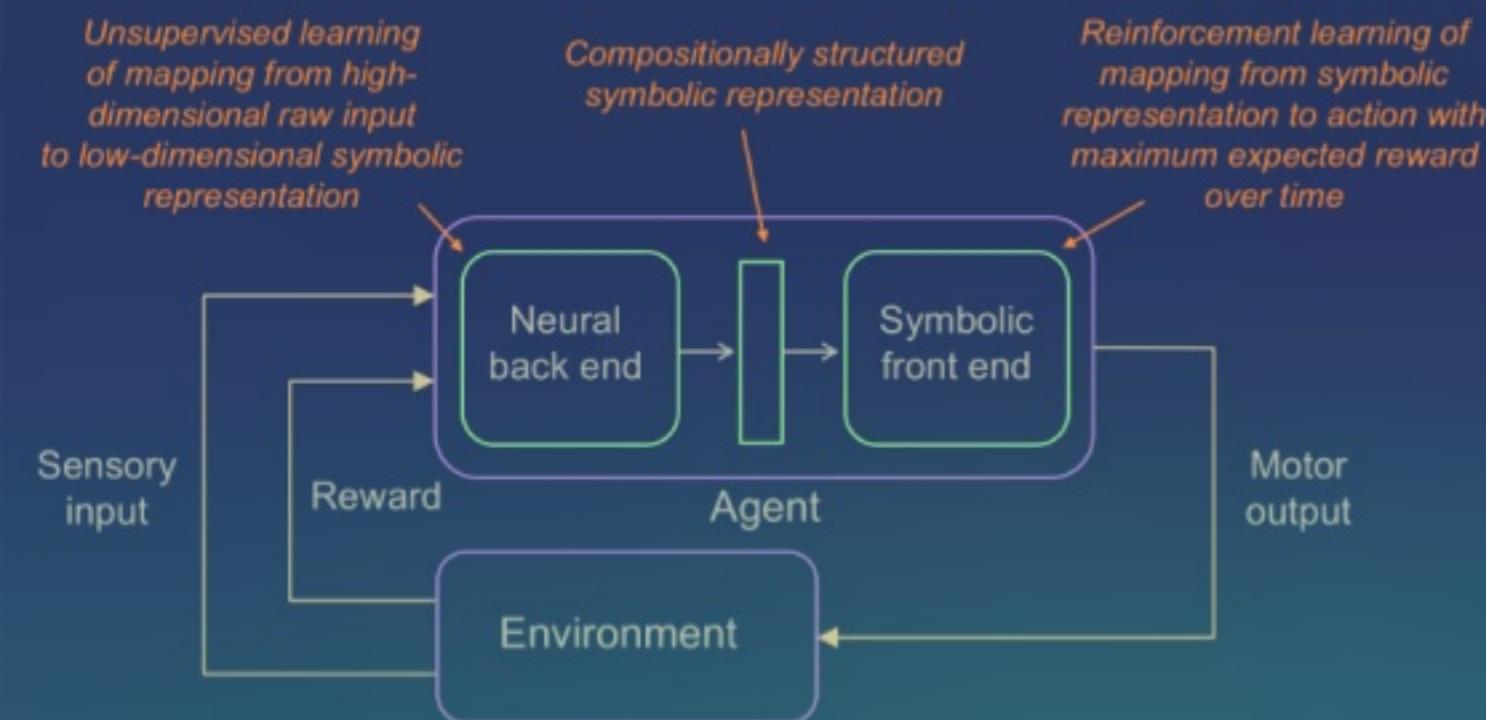
AGI - Machine Learning, Language, Reasoning, Robotics

Planet Earth <http://wiki.opencog.org> opencog@googlegroups.com

Repositories People 23



Deep Symbolic Reinforcement Learning



Towards Deep Symbolic Reinforcement Learning

Marta Garnelo
Department of Computing
Imperial College London
London, SW7 2AZ
garnelo@ic.ac.uk

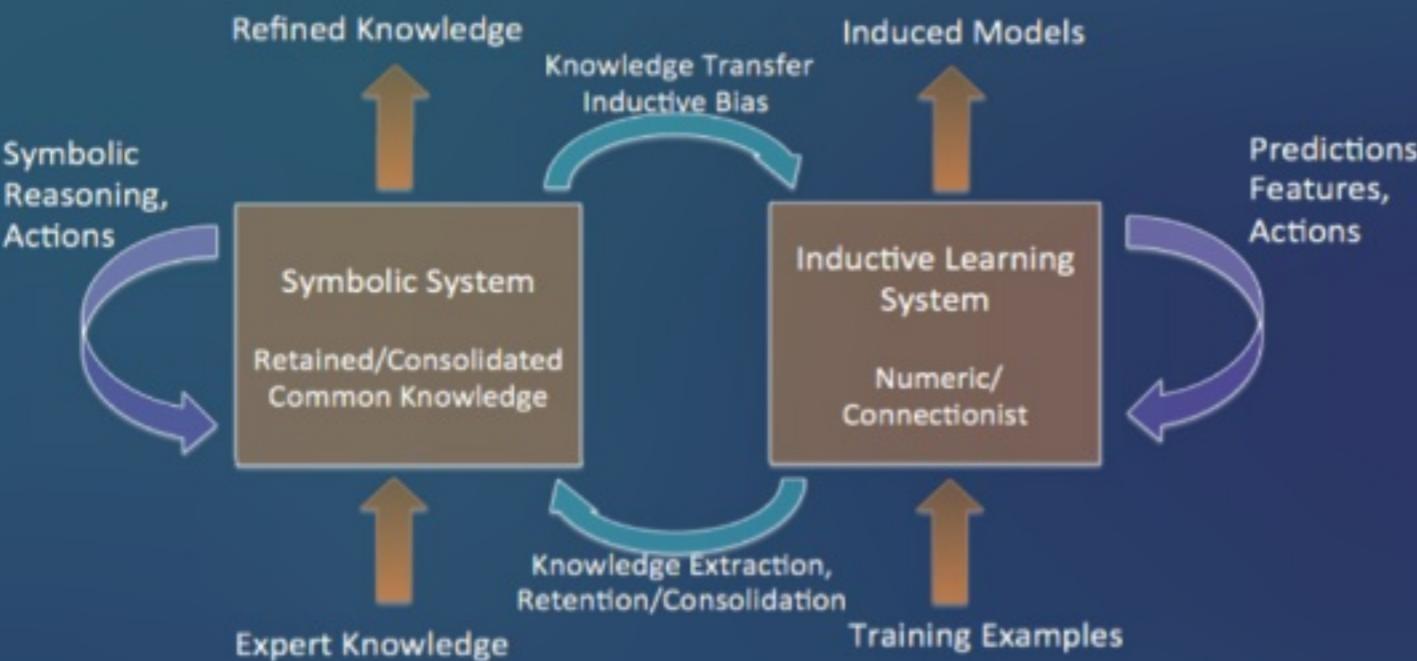
Kai Arulkumaran
Department of Bioengineering
Imperial College London
London, SW7 2AZ
kailash.arulkumaran13@ic.ac.uk

Murray Shanahan
Department of Computing
Imperial College London
London, SW7 2AZ
m.shanahan@ic.ac.uk

Abstract

Deep reinforcement learning (DRL) brings the power of deep neural networks to bear on the generic task of trial-and-error learning, and its effectiveness has been convincingly demonstrated on tasks such as Atari video games and the game of

LML: Lifelong Machine Learning



On Common Ground: Neural-Symbolic Integration and Lifelong Machine Learning

Daniel L. Silver
Acadia University
Nova Scotia, Canada
danny.silver@acadiau.ca

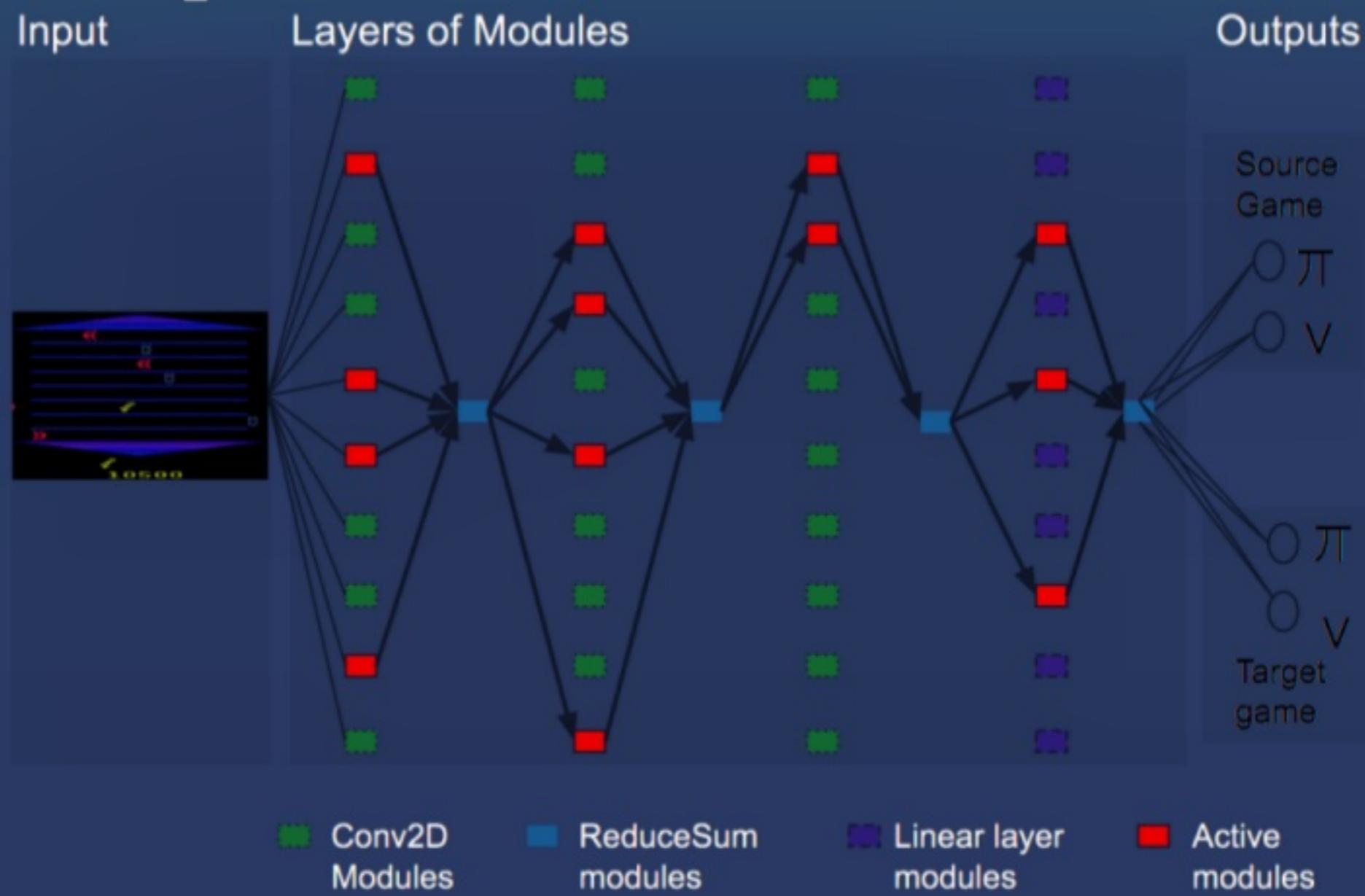
Abstract

Research efforts in Neural-Symbolic Integration and Lifelong Machine Learning have taken place with limited interactions over the last 20 years. These two areas of artificial intelligence share common ground and yet have much to learn from each other. This paper provides background, particularly on Lifelong Machine Learning, and presents a number of common areas of investigation with Neural-Symbolic Integration. It then invites re-

and connectionist paradigms of artificial intelligence for both learning and reasoning [Garcez *et al.*, 2002; Garcez and Gabbay, 2004].

NSI seeks to make use of the learning capacities of neural network models and the reasoning capacities of logic [Garcez *et al.*, 2014]. In a NSI system, neural networks provide the machinery for parallel computation and robust learning, while symbolic logic provides knowledge representation and reasoning. The retention of symbolic knowledge of learned models can be used for transfer learning, explanation of the models to humans, and use of the knowledge by other sys-

DeepMind's PathNet

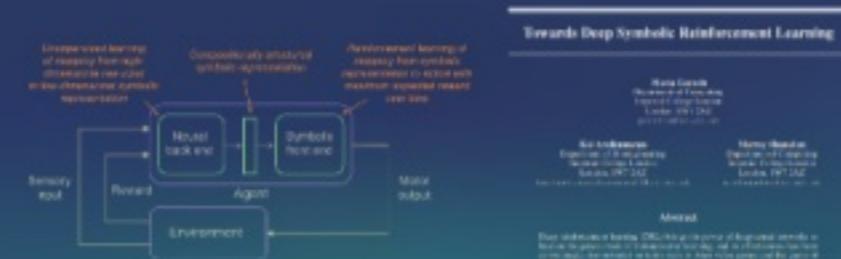


Towards AGI

OpenCog



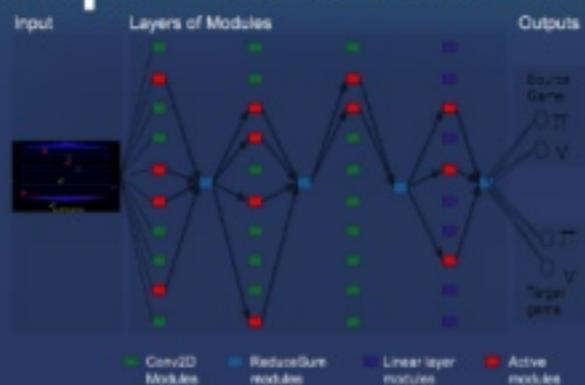
Deep Symbolic Reinforcement Learning



LML: Lifelong Machine Learning

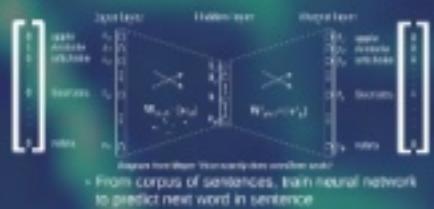


DeepMind's PathNet

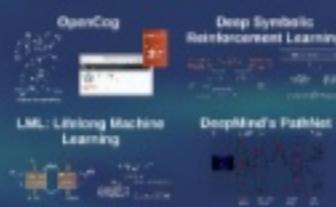


Examples

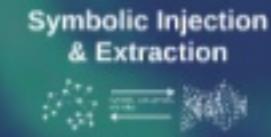
word2vec



Towards AGI



Enhance neural with symbolic



Enhance symbolic with neural



Pipelines



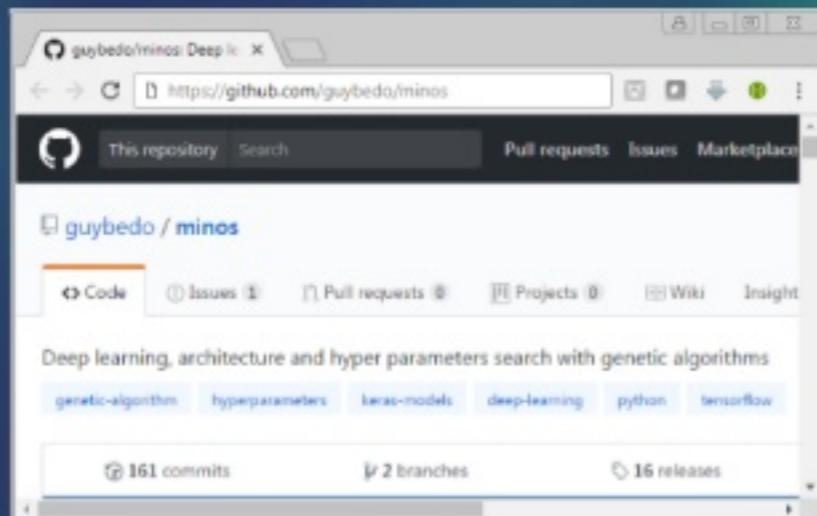
Understanding



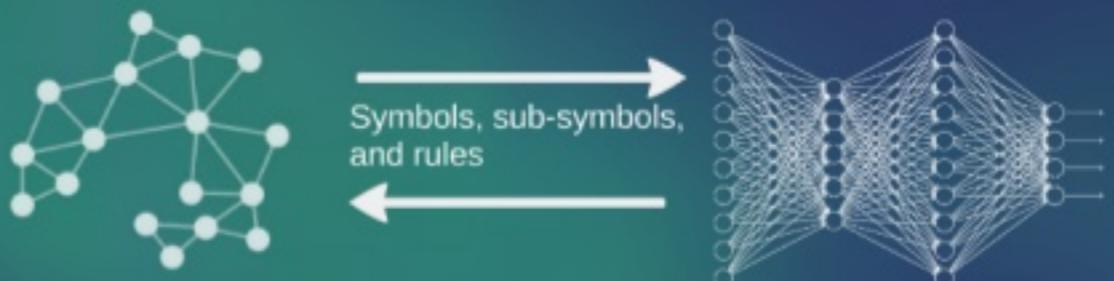
Explaining the "black box" of a neural network
Image from survivingmachine.com

Enhance neural with symbolic

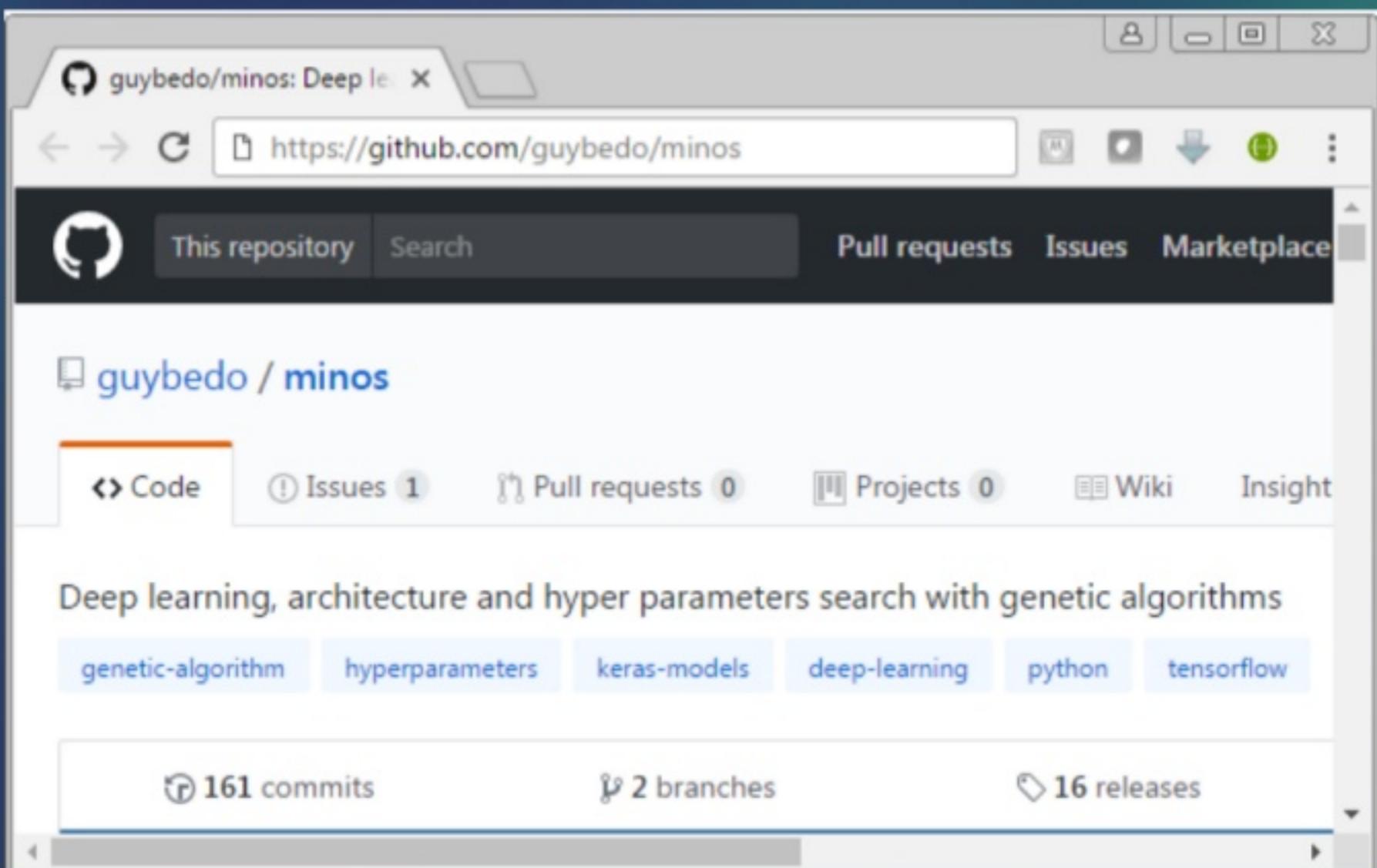
Minos



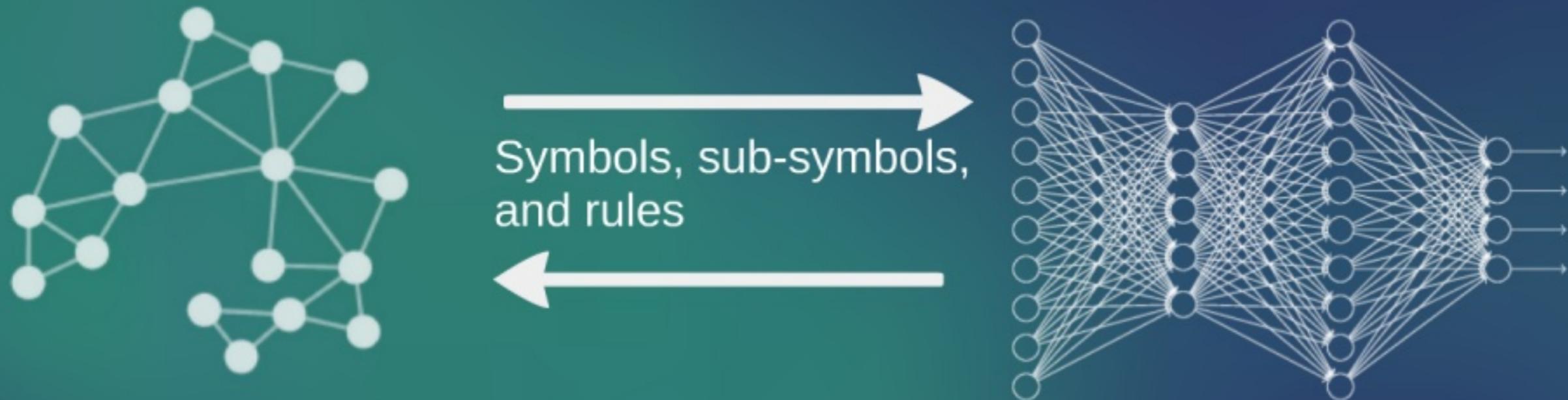
Symbolic Injection & Extraction



Minos

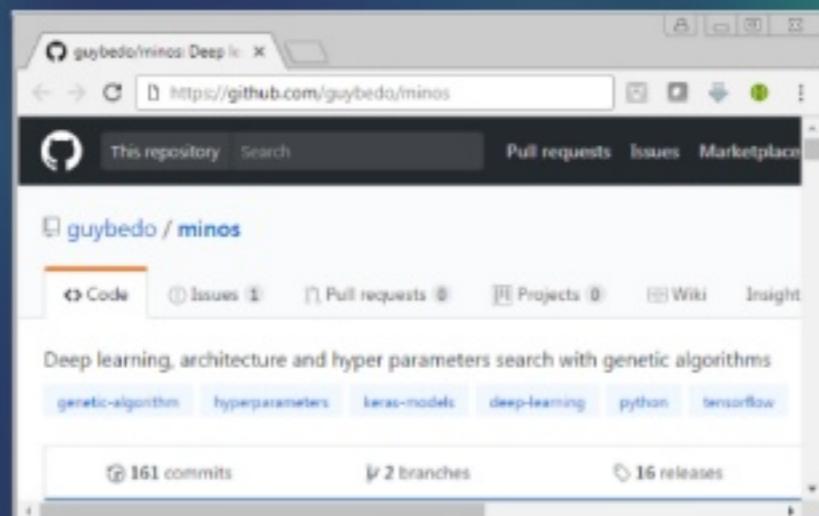


Symbolic Injection & Extraction

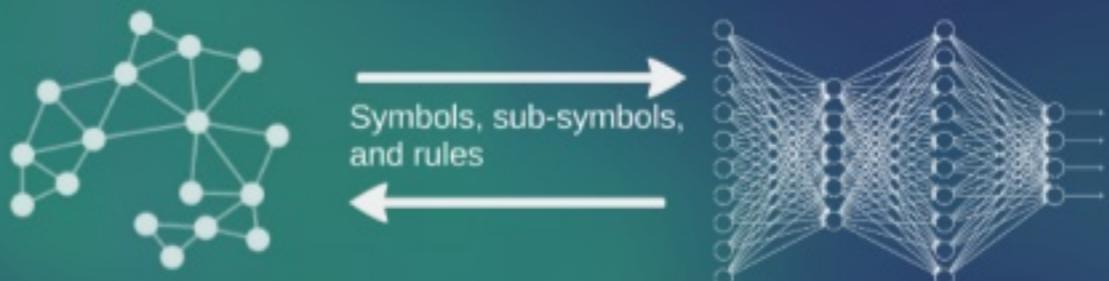


Enhance neural with symbolic

Minos

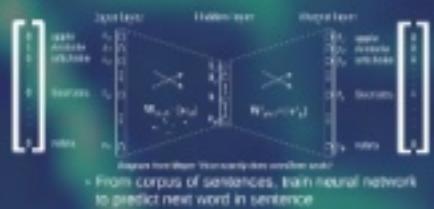


Symbolic Injection & Extraction



Examples

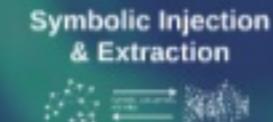
word2vec



Towards AGI



Enhance neural with symbolic



Enhance symbolic with neural



Pipelines



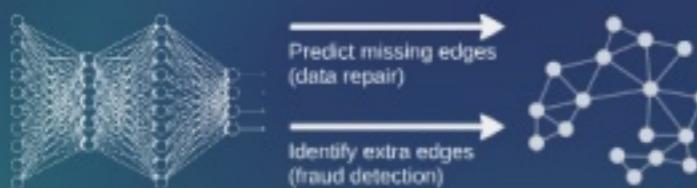
Understanding



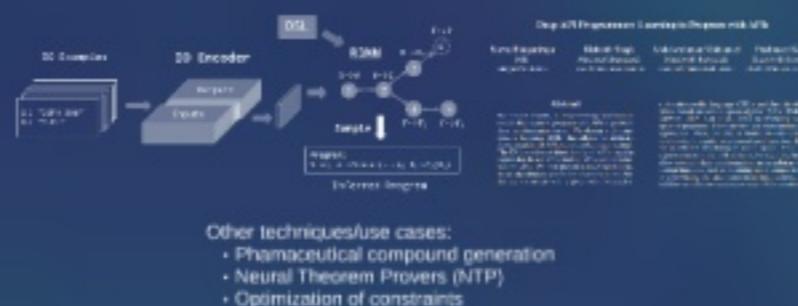
- Transfer Learning
- One Shot Learning

Enhance symbolic with neural

Neural predicts for knowledge graph



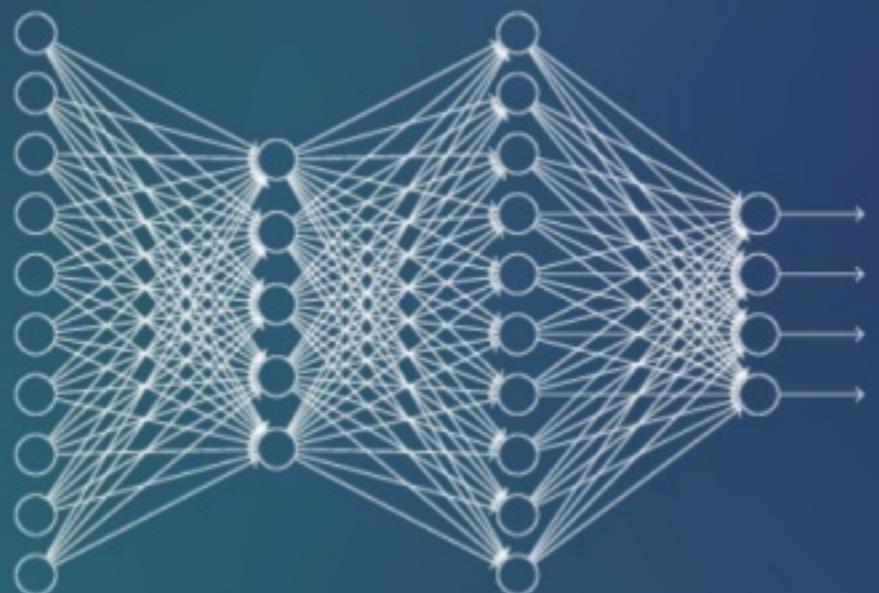
Neural-guided search



Knowledge Graph Embedding



Neural predicts for knowledge graph



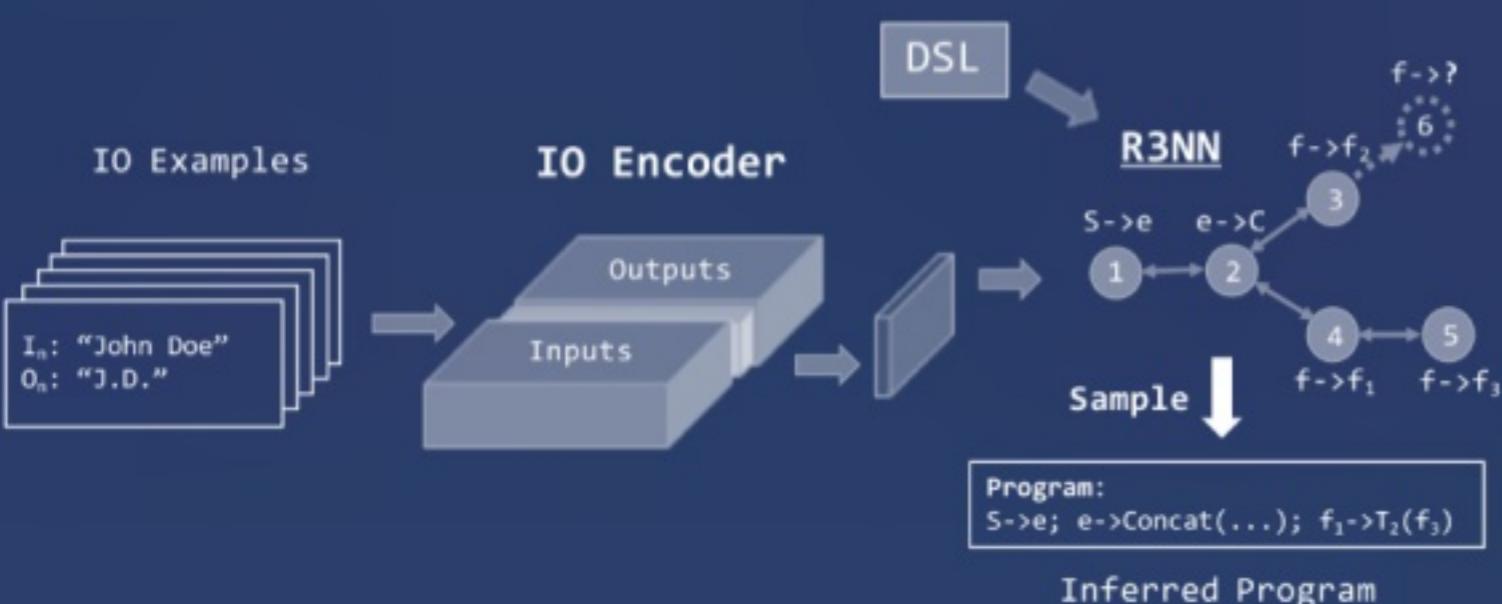
Predict missing edges
(data repair)



Identify extra edges
(fraud detection)



Neural-guided search



Deep API Programmer: Learning to Program with APIs

Surya Bhupatiraju
MIT
surya@mit.edu

Rishabh Singh
Microsoft Research
risin@microsoft.com

Abdel-rahman Mohamed
Microsoft Research
asamir@microsoft.com

Pushmeet Kohli
Microsoft Research
pkohli@microsoft.com

Abstract

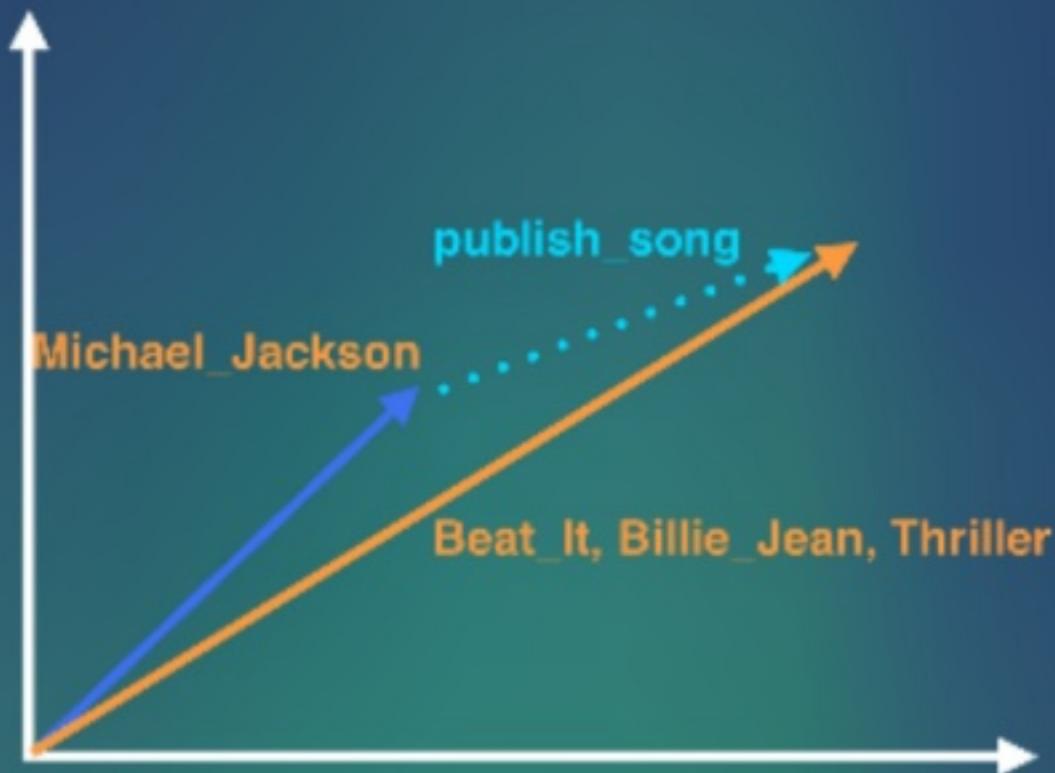
We present DAPIP, a Programming-By-Example system that learns to program with APIs to perform data transformation tasks. We design a domain-specific language (DSL) that allows for arbitrary concatenations of API outputs and constant strings. The DSL consists of three family of APIs: regular expression-based APIs, lookup APIs, and transformation APIs. We then present a novel neural synthesis algorithm to search for programs in the DSL that are consistent with a given set of examples.

a domain-specific language (DSL), and then develop algorithms based on version-space algebra (VSA) [Položay and Gulwani, 2015; Lau *et al.*, 2003] to efficiently search the space of programs. There are two key shortcomings of these approaches. First, the DSL is limited to only certain low-level syntactic regular expression-based operators that allow for an efficient structuring of search space. This limits the expressiveness of the PBE systems; for example, they do not allow *semantic* data transformations using arbitrary transformation functions such as obtaining month names from a date or abbreviating the state name in an input address. Second, building an efficient synthesizer using VSA requires a large

Other techniques/use cases:

- Pharmaceutical compound generation
- Neural Theorem Provers (NTP)
- Optimization of constraints

Knowledge Graph Embedding

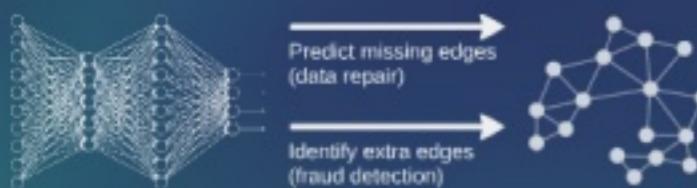


- Encode graph relations as vectors
- Convenient input to neural networks

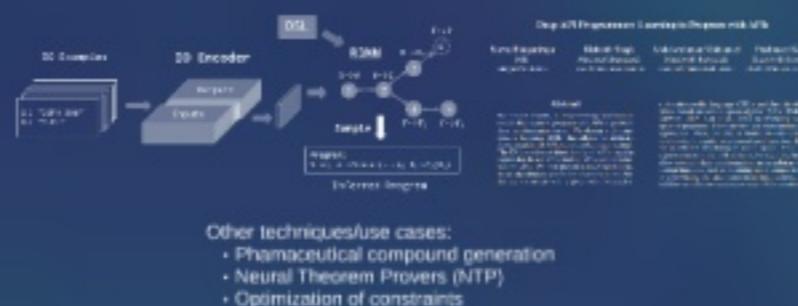
Image from Feng et al "Knowledge Graph Embedding by Flexible Translation"

Enhance symbolic with neural

Neural predicts for knowledge graph



Neural-guided search

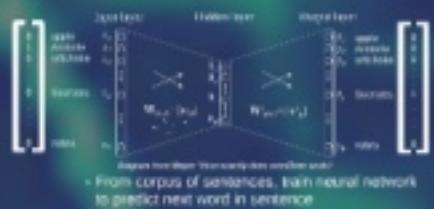


Knowledge Graph Embedding



Examples

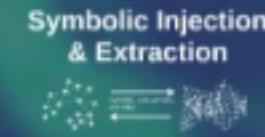
word2vec



Towards AGI



Enhance neural with symbolic



Enhance symbolic with neural



Pipelines



Understanding

Explaining the "black box" of a neural network



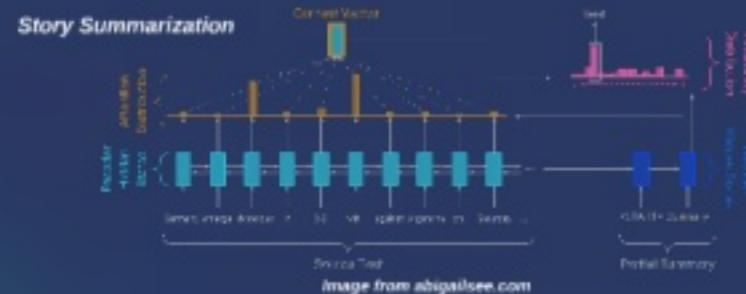
Image from survivingmachine.com

Transfer Learning

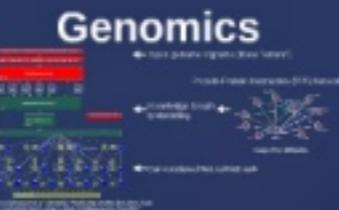
One-Shot Learning

Pipelines

Sequence-to-Sequence



Sequence-to-Class



Neural-to-Symbolic

Self-Driving Cars

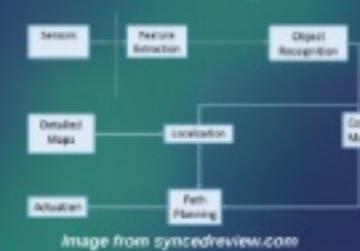


Image from syncedreview.com

- Computer Vision
 - Face Recognition
 - Security Cameras

Sequence-to-Sequence

Story Summarization

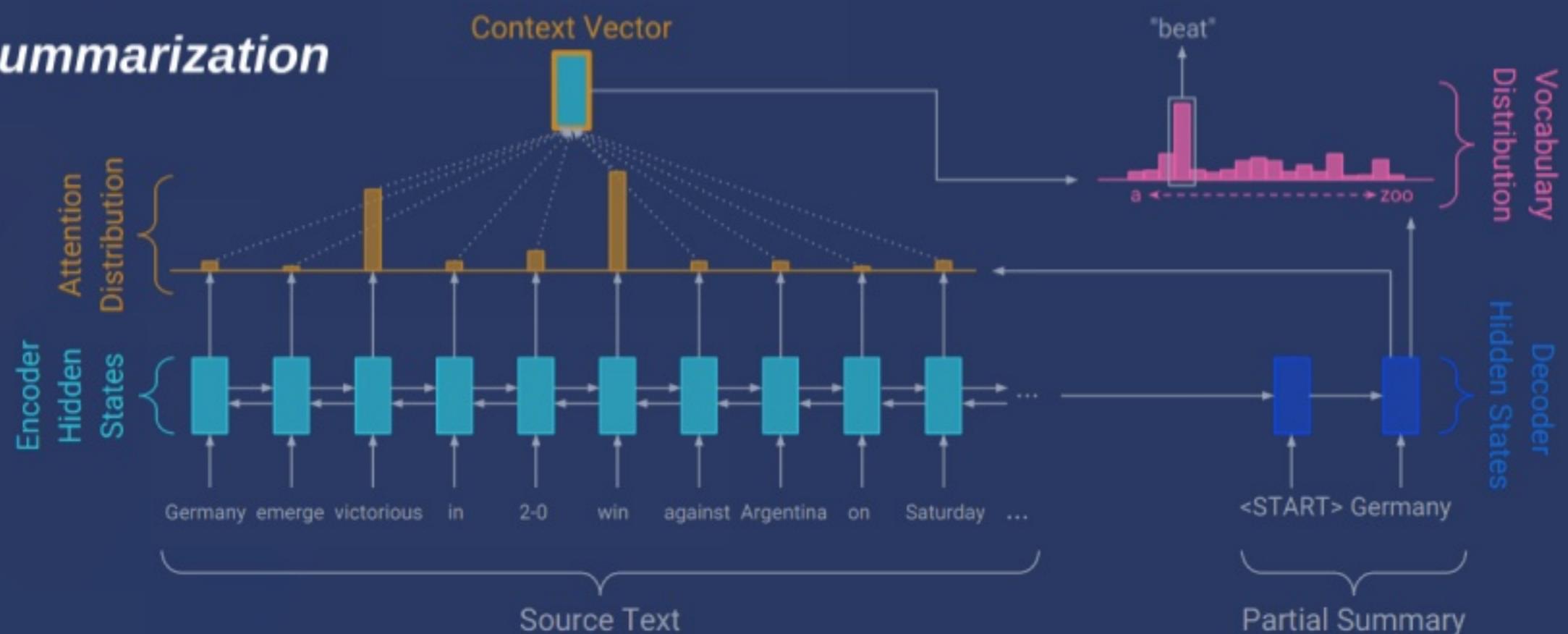


Image from abigailsee.com

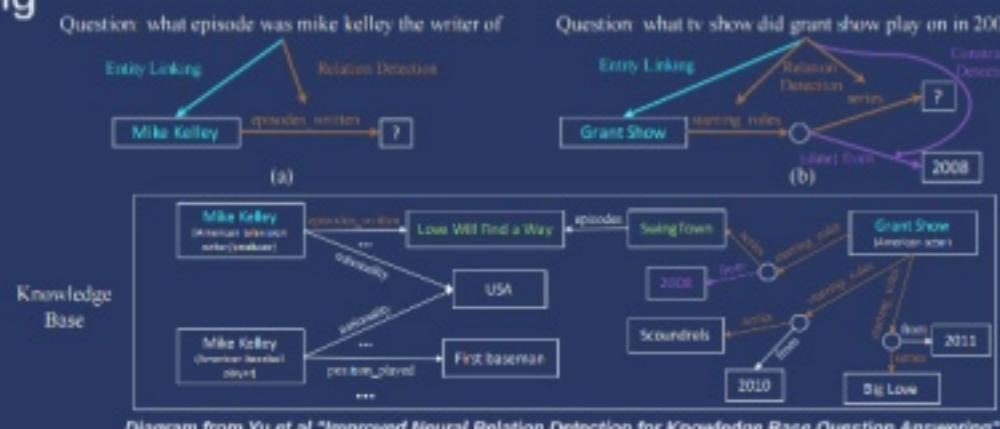
Other techniques/use cases:

- Human language translation
- Code generation from natural language requirements

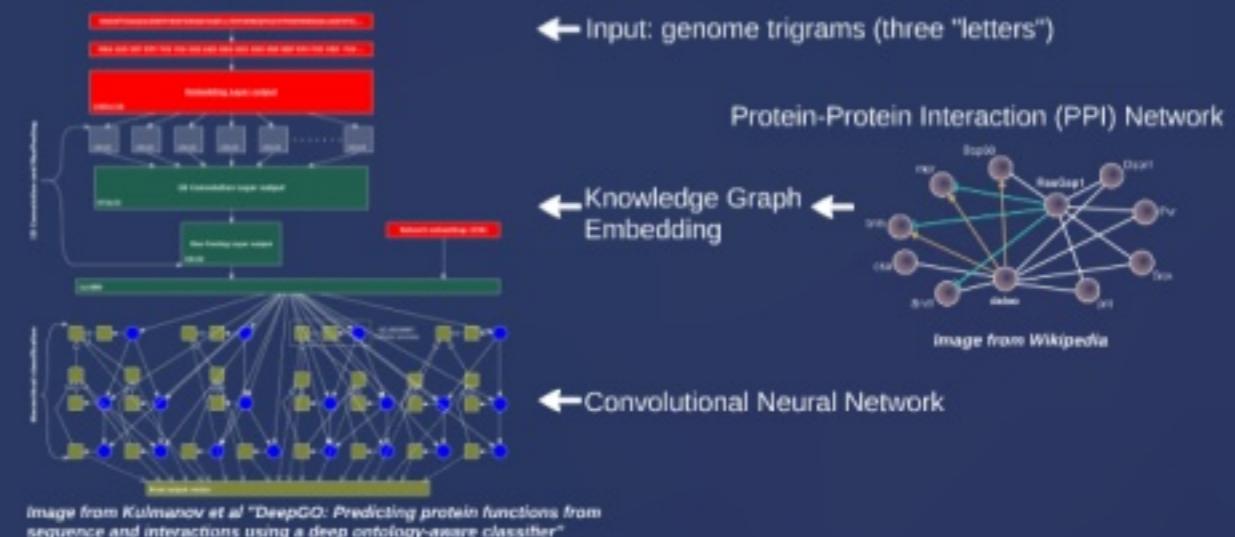
Sequence-to-Class

NLP

- Sentiment Analysis
- Parsing



Genomics



NLP

- Sentiment Analysis
- Parsing

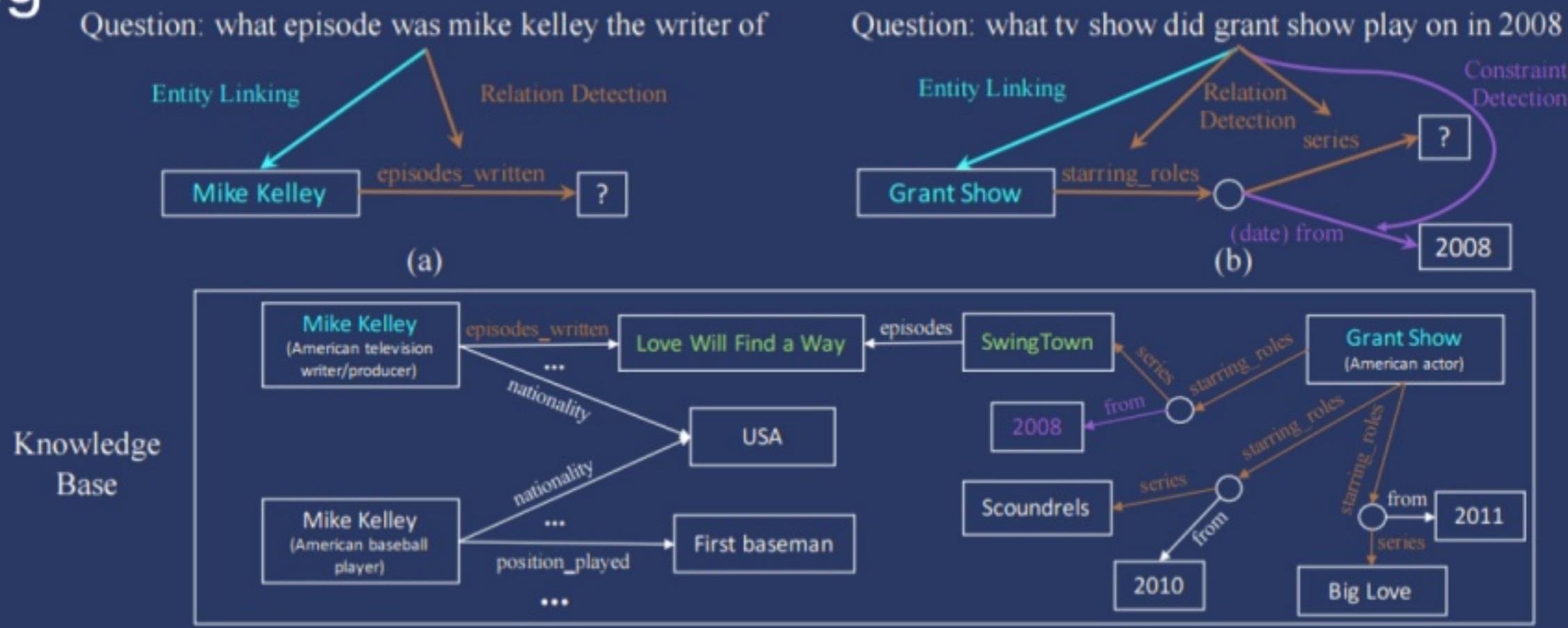
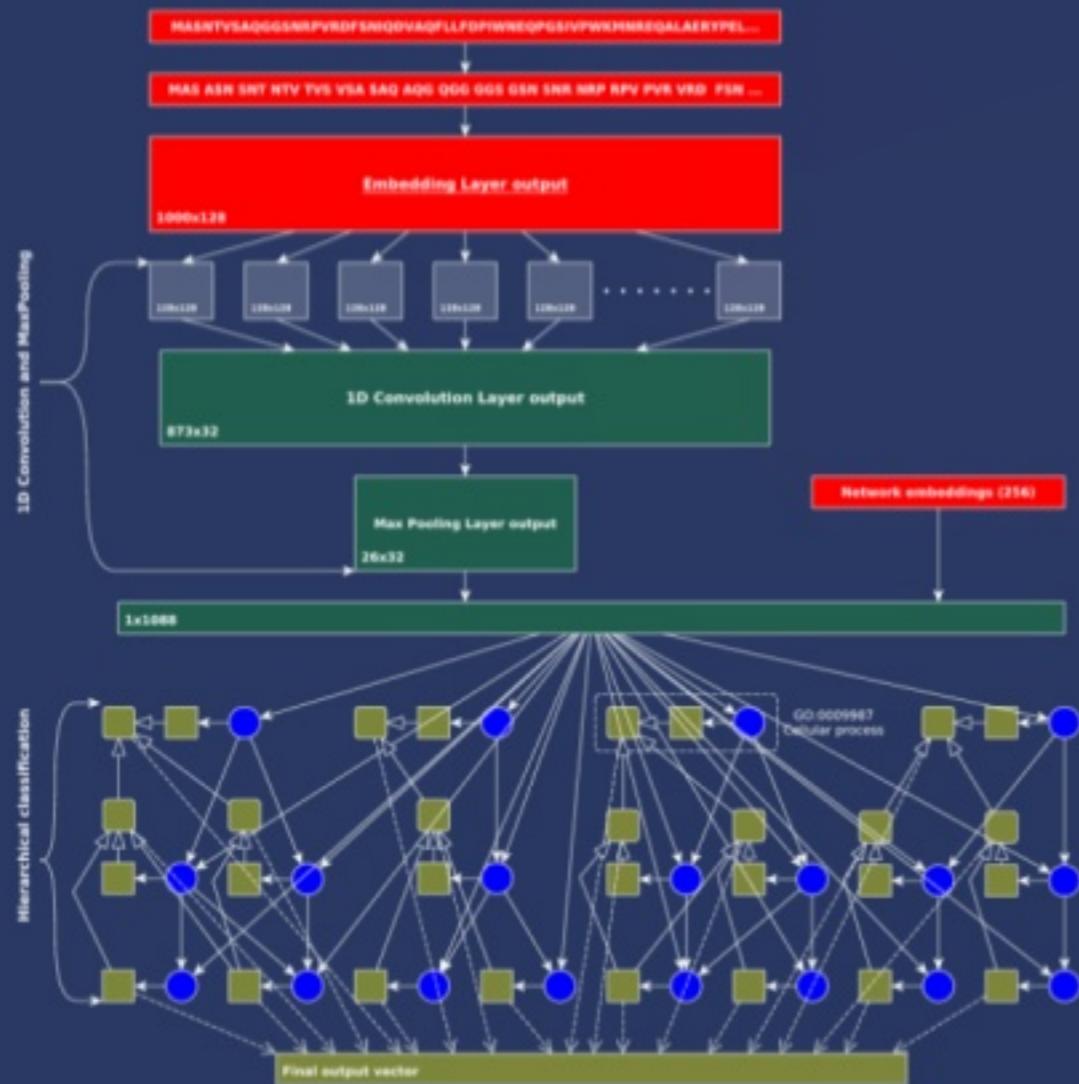


Diagram from Yu et al "Improved Neural Relation Detection for Knowledge Base Question Answering"

Genomics



← Input: genome trigrams (three "letters")

Protein-Protein Interaction (PPI) Network

← Knowledge Graph Embedding

← Convolutional Neural Network

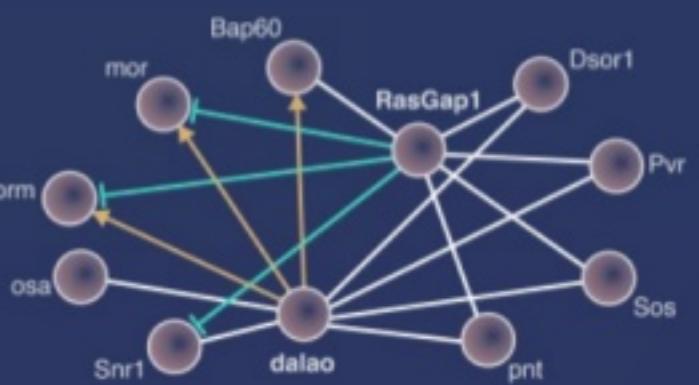


Image from Wikipedia

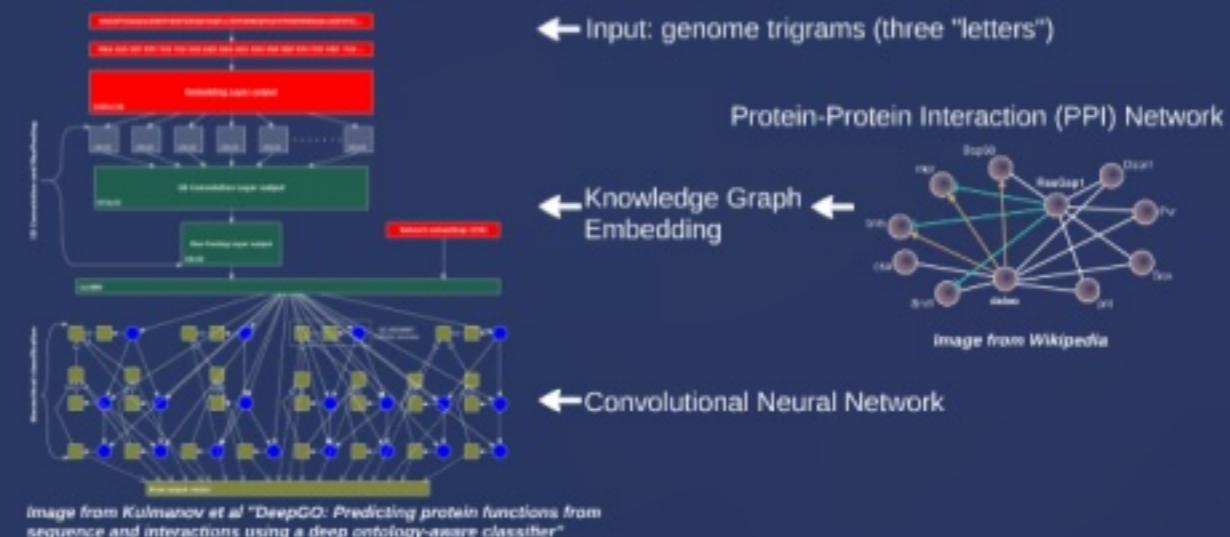
Image from Kulmanov et al "DeepGO: Predicting protein functions from sequence and interactions using a deep ontology-aware classifier"

Sequence-to-Class

NLP

- Sentiment Analysis
 - Parsing

Genomics



Neural-to-Symbolic

- Self-Driving Cars

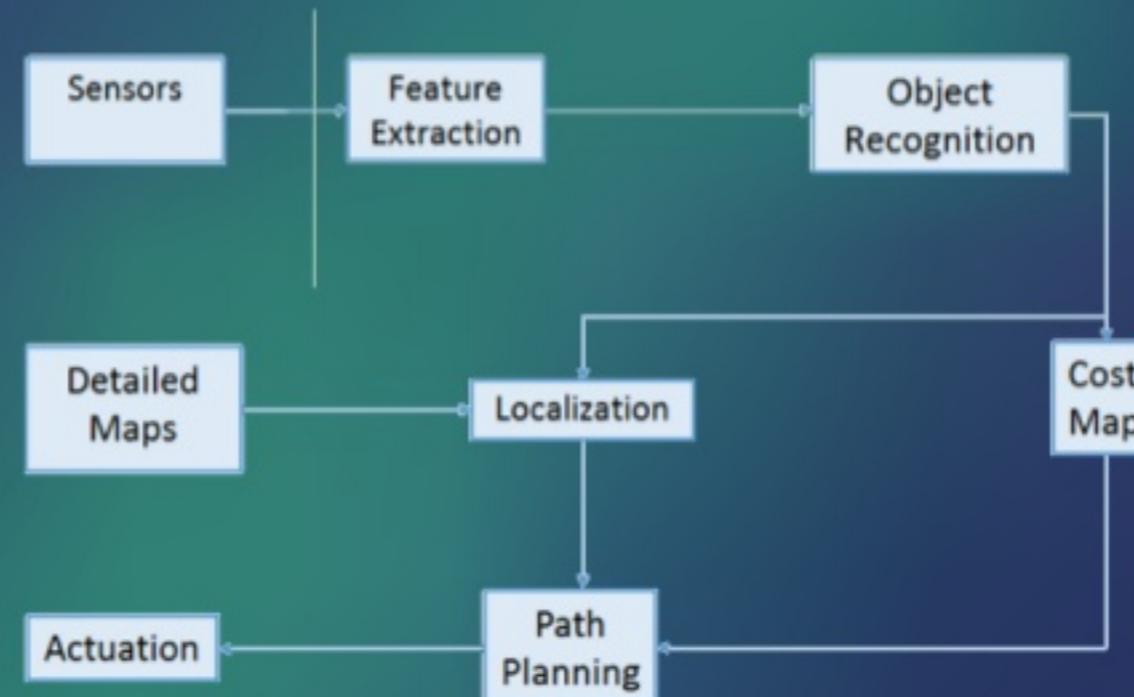
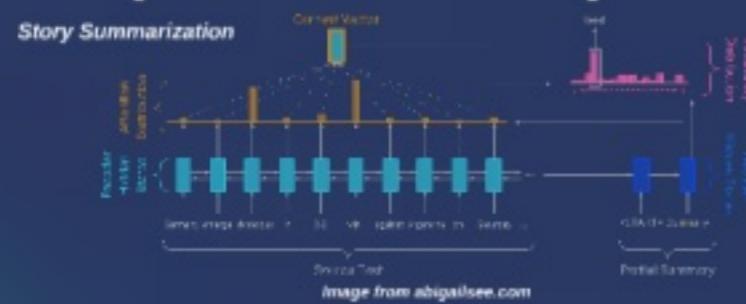


Image from syncedreview.com

- Computer Vision
 - Face Recognition
 - Security Cameras

Pipelines

Sequence-to-Sequence



Other techniques/use cases:
• Human language translation
• Code generation from natural language requirements

Sequence-to-Class



Neural-to-Symbolic

Self-Driving Cars

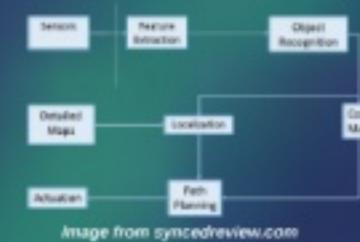


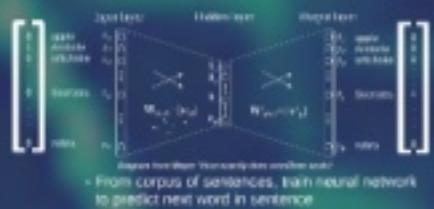
Image from syncerreview.com

Computer Vision

- Face Recognition
- Security Cameras

Examples

word2vec



Towards AGI

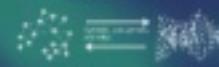


Enhance neural with symbolic

Minos



Symbolic Injection & Extraction



Enhance symbolic with neural

Neural predicts for knowledge graph



Neural-guided search



Knowledge Graph Embedding



Pipelines



Understanding

Explaining the "black box" of a neural network



Transfer Learning
One Shot Learning

Understanding

- Explaining the "black box" of a neural network

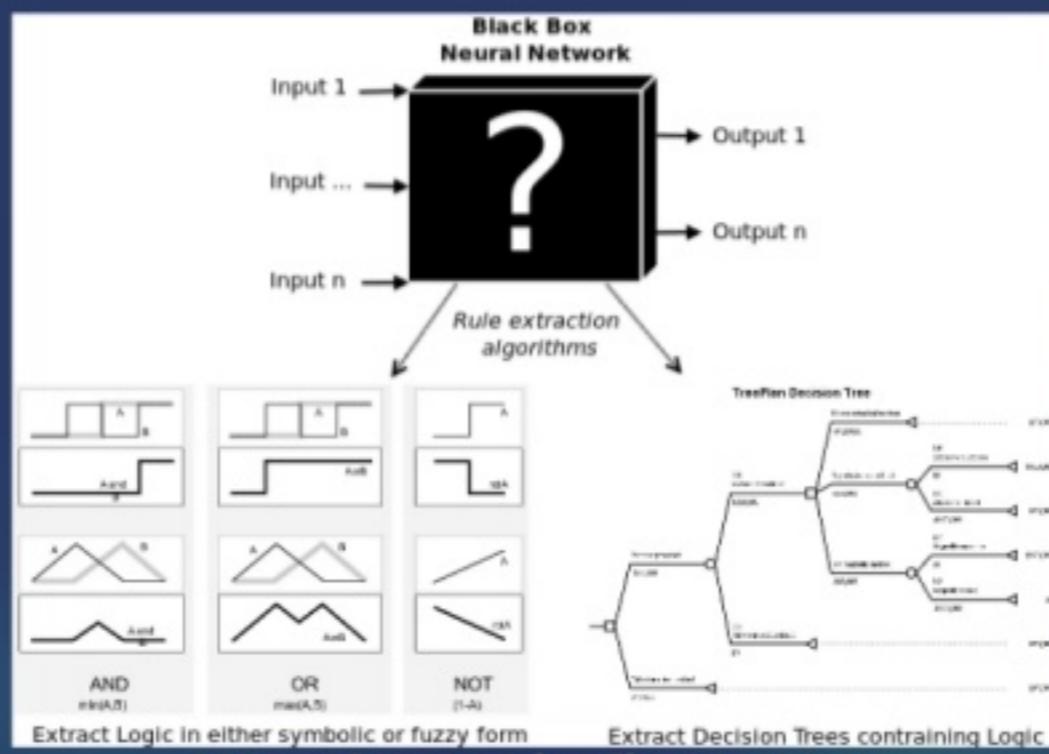
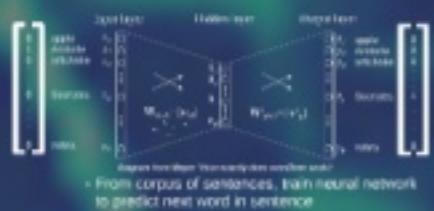


Image from turingfinance.com

- Transfer Learning
 - One-Shot Learning

Examples

word2vec



Towards AGI



Enhance neural with symbolic



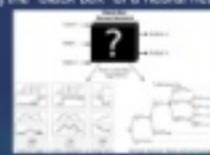
Enhance symbolic with neural



Pipelines



Understanding



Explaining the "black box" of a neural network

Image from survivingmachine.com

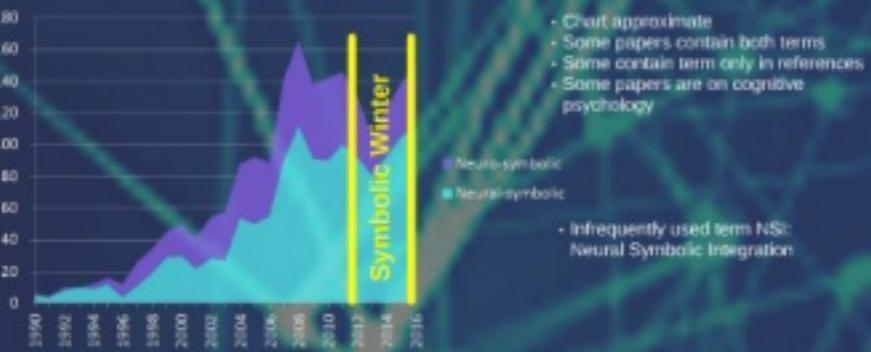
Transfer Learning
One Shot Learning

What It's Not

- All systems have symbols as input and output
 - LSTM (long short-term memory)
 - NTM (Neural Turing Machines)

Neuro-Symbolic

Trends



Examples



What It's Not

- All systems have symbols as input and output
- LSTM (long short-term memory)
- NTM (Neural Turing Machines)

Philosophy

What is AI?

Neuro-Symbolic

Weak AI
Classic AI



Strong AI



Artificial General Intelligence (AGI)

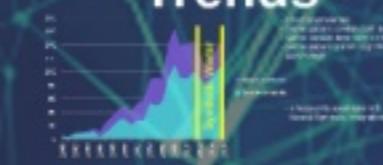
Artificial Super Intelligence (ASI)

It's OK to say "AI" now

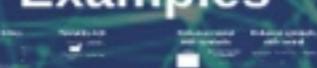
AI Permafrost

Silicon Valley AI Index

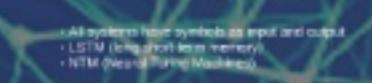
Trends



Examples

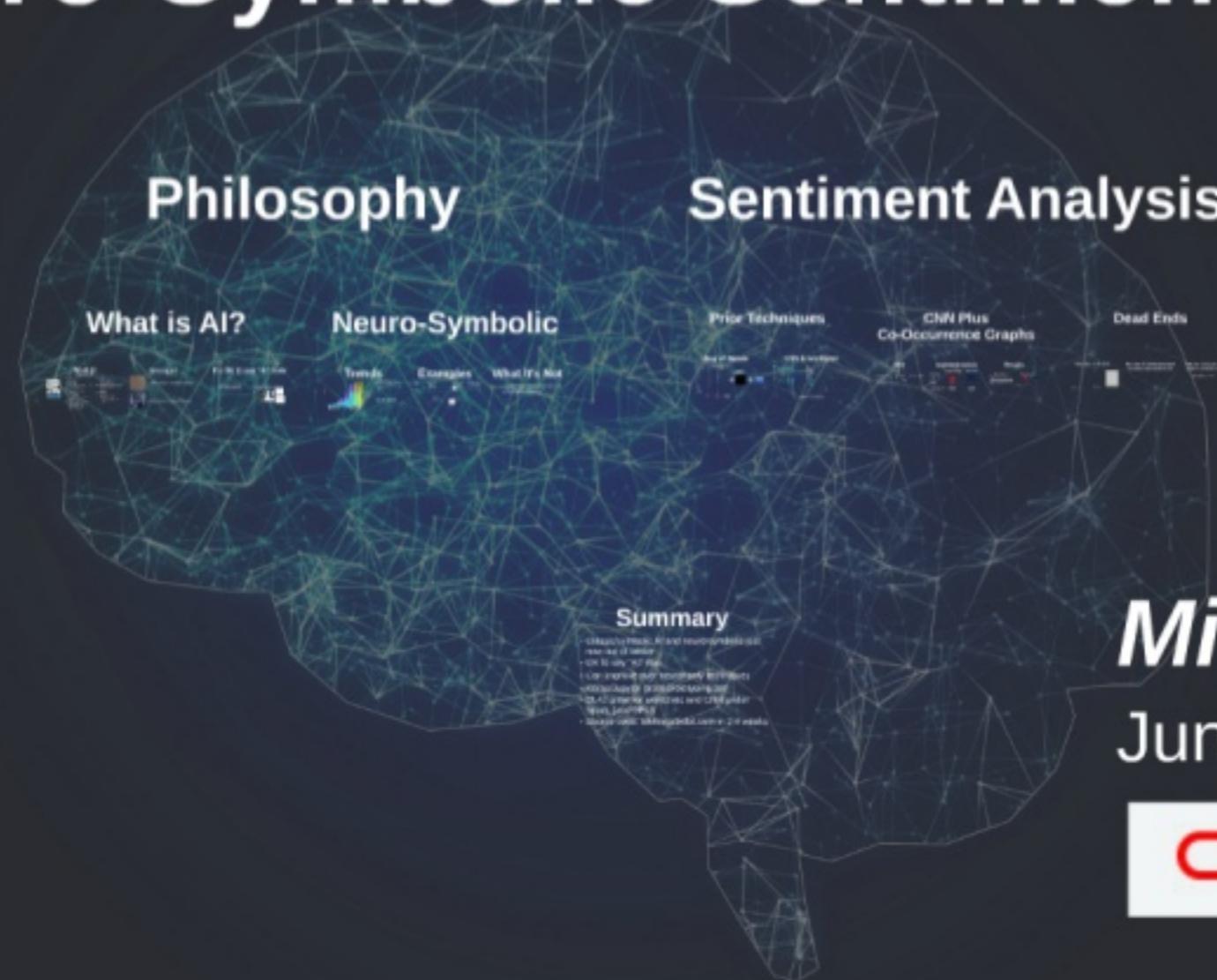


What It's Not



- AI systems have symbols as input and output
- LSTM (long short-term memory)
- NTM (Neural Turing Machines)

Neuro-Symbolic Sentiment Analysis



Michael Malak

June 7, 2017

ORACLE® Cloud

Big Data Preparation

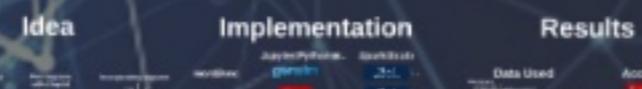
Analytics Cloud

Sentiment Analysis

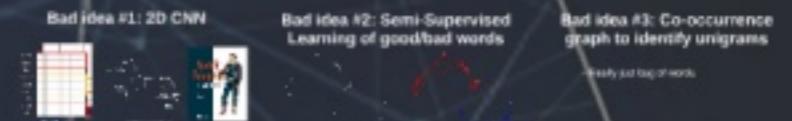
Prior Techniques



CNN Plus Co-Occurrence Graphs

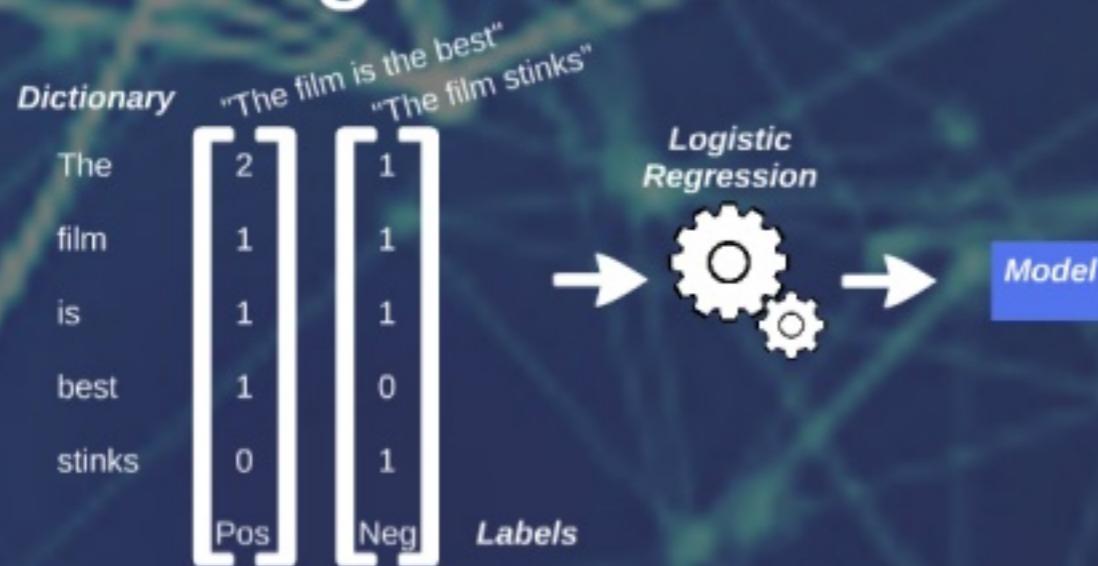


Dead Ends

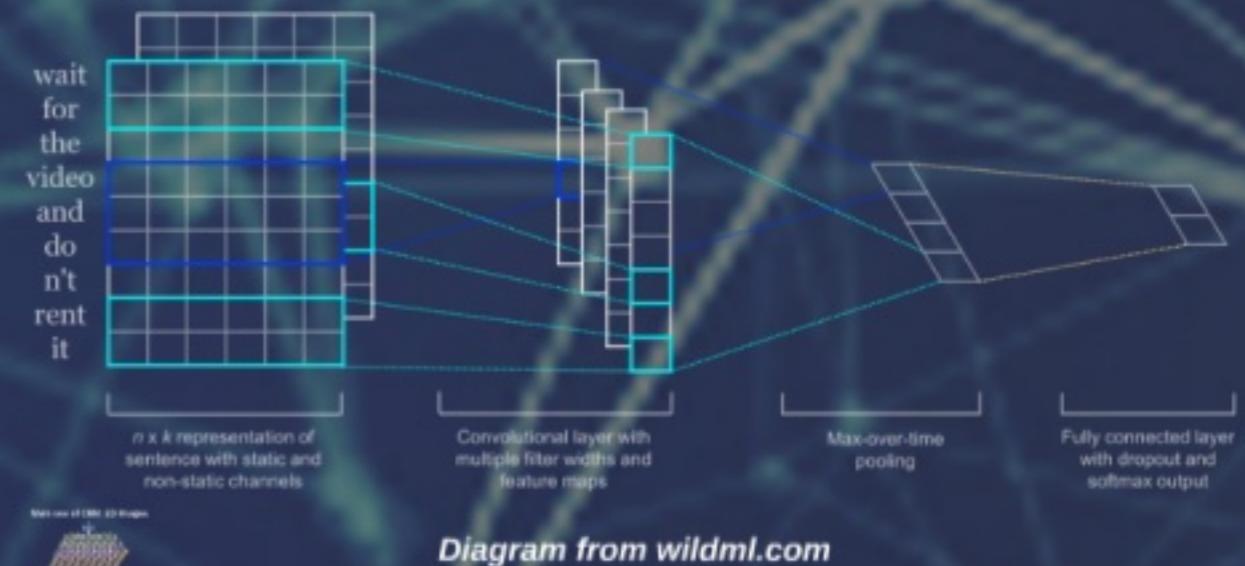


Prior Techniques

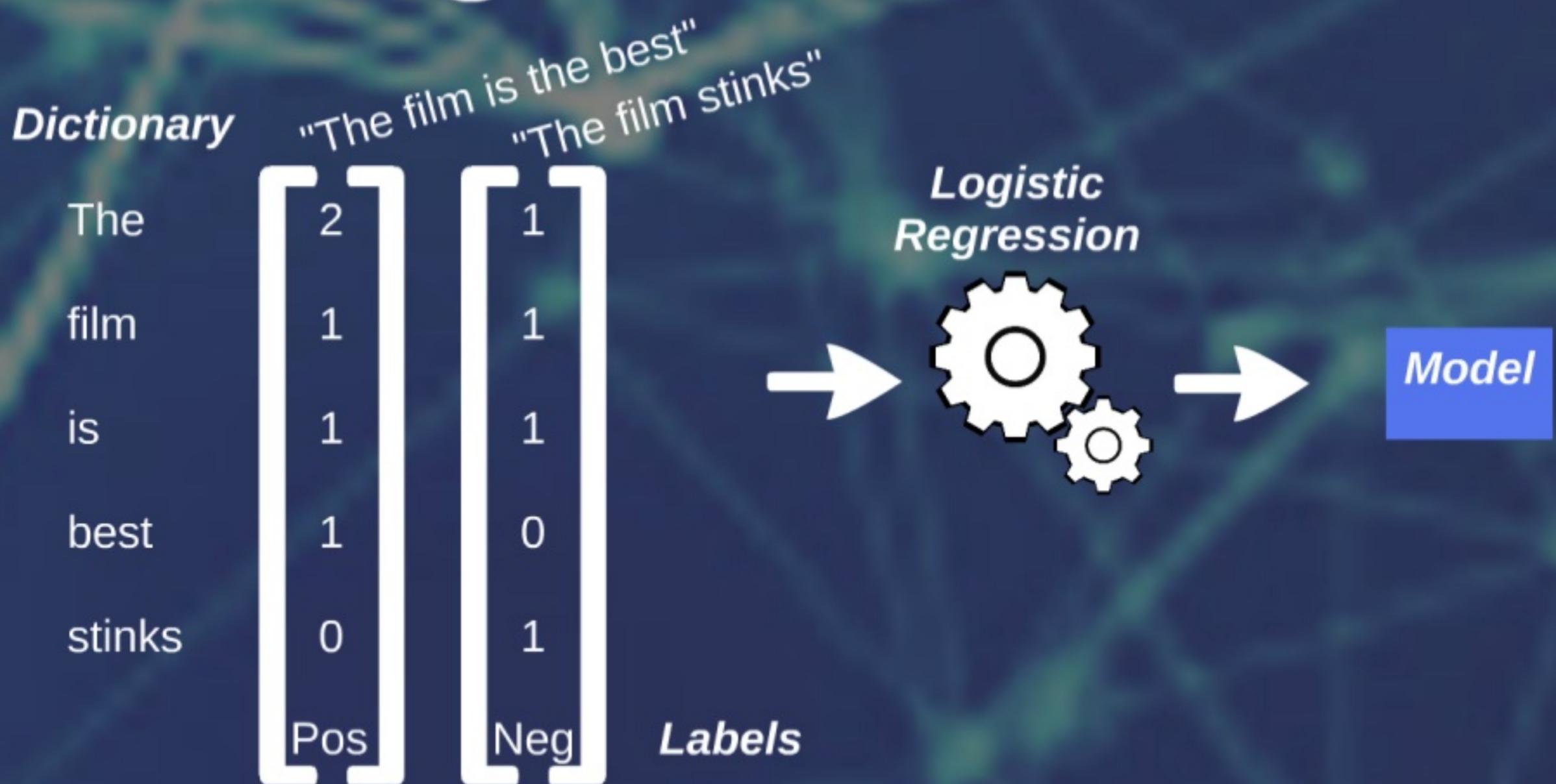
Bag of Words



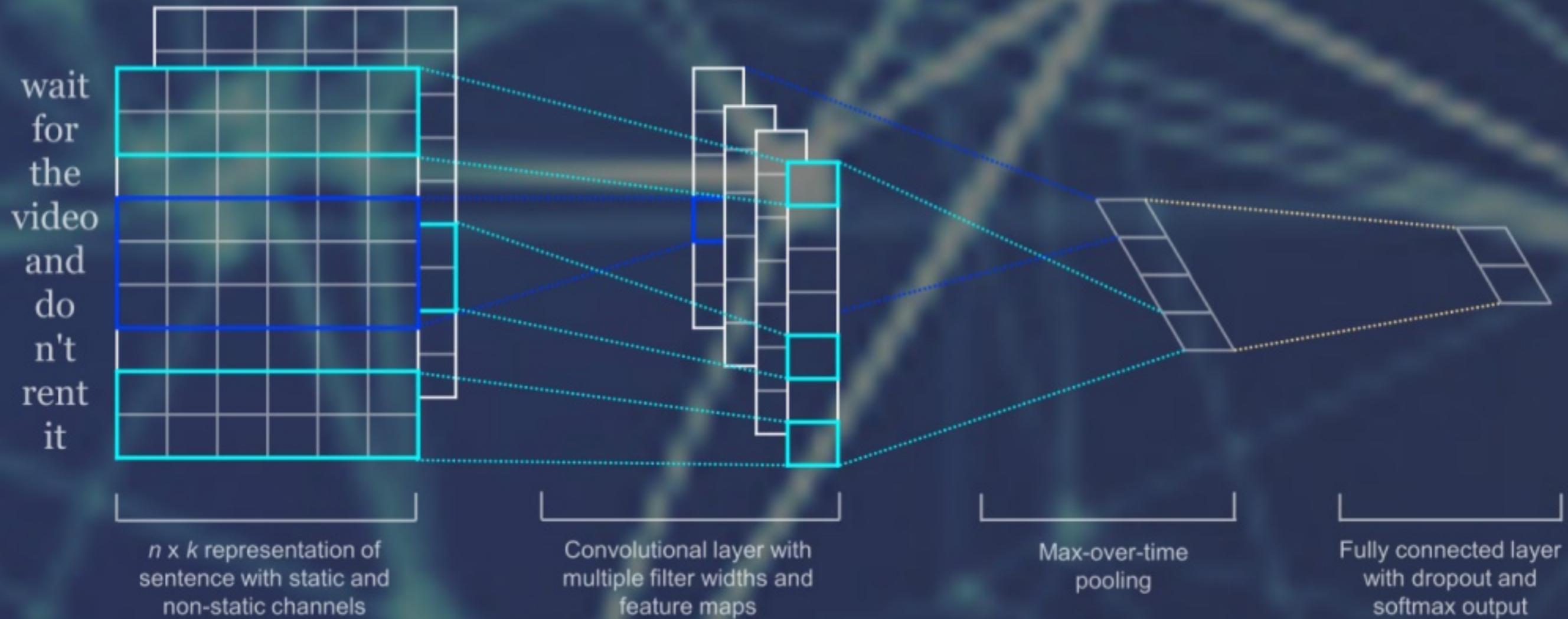
CNN & word2vec



Bag of Words



CNN & word2vec



Main use of CNN: 2D images



Diagram from wildml.com

Main use of CNN: 2D Images

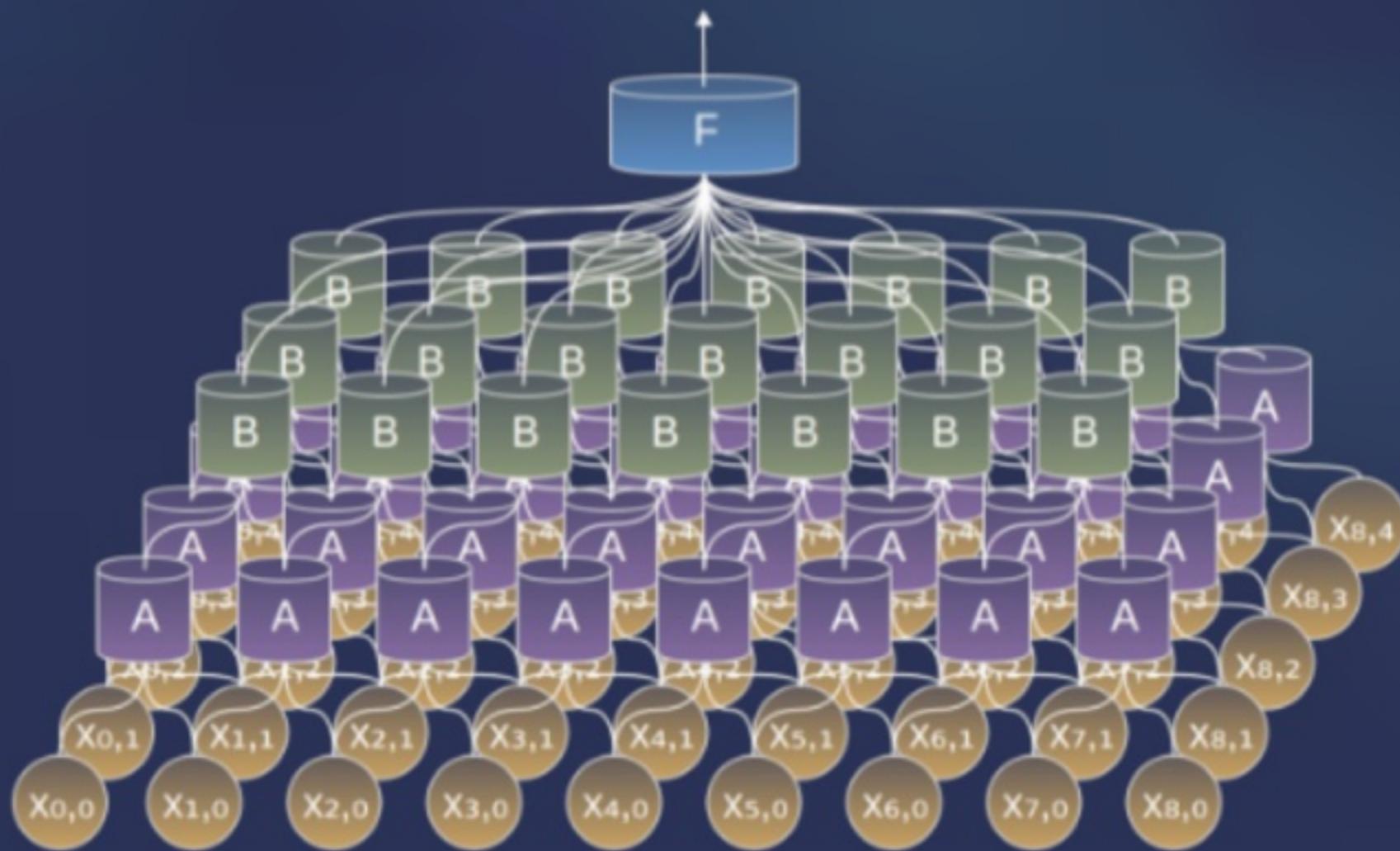
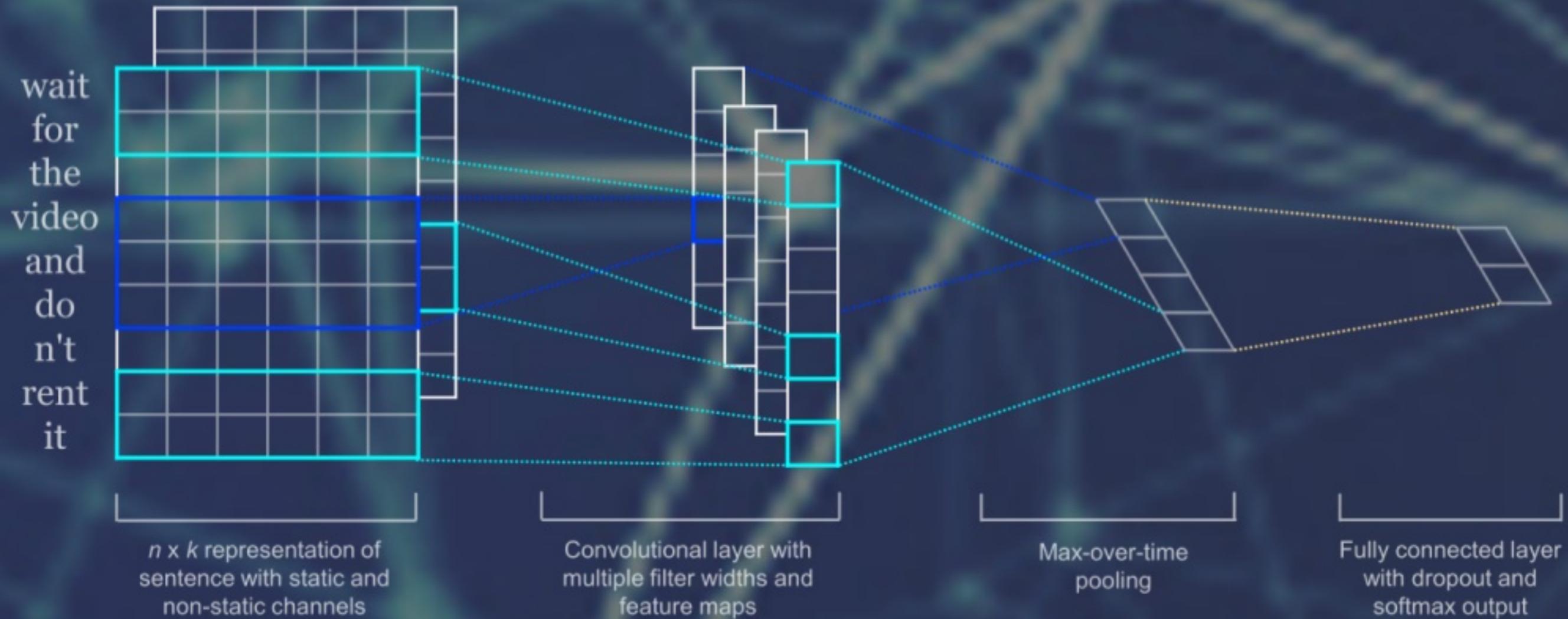


Image from colah.github.io

CNN & word2vec



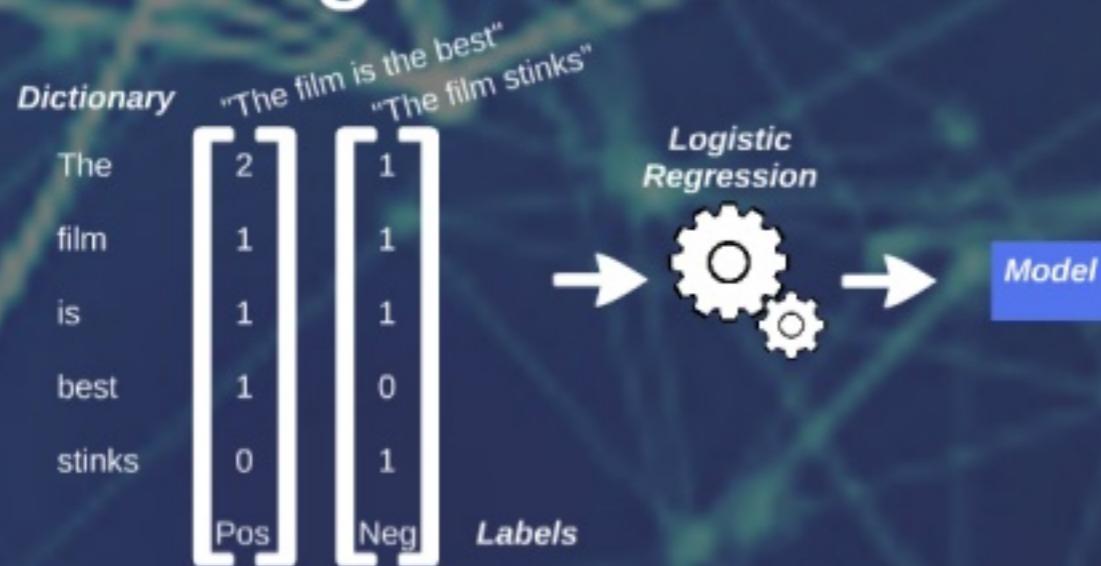
Main use of CNN: 2D images



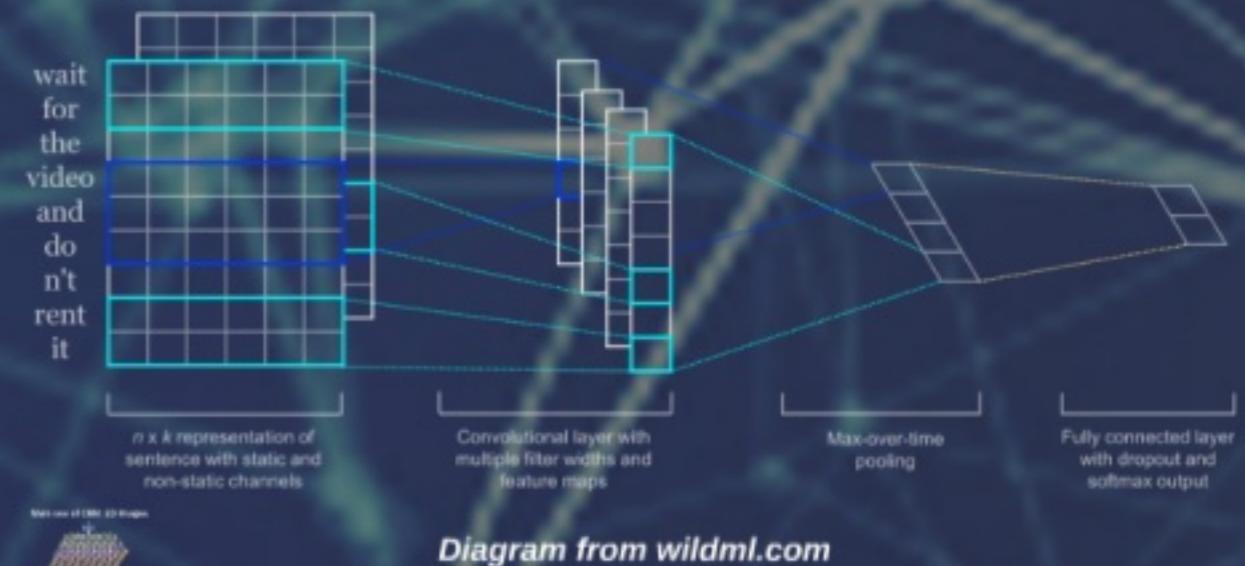
Diagram from wildml.com

Prior Techniques

Bag of Words



CNN & word2vec



CNN Plus Co-Occurrence Graphs

Idea



Implementation



Results



Idea

Bigrams carry context

Sentiment	
"a great work"	Pos
"a great piece"	Pos
"a piece of work"	Neg

("piece","work") is a bigram with negative sentiment



Diagram from Richter et al "Exploiting the Leipzig corpora collection"

Find bigrams with GraphX

- If two words appear near each other in the training corpus, draw an edge
- (or increment edge count if there is already an edge)
- Edges with high count are the important bigrams to track

Incorporating bigrams

	word2vec vectors (300 columns)				bigram presence indicators (300 columns)			
	The	film	is	a	piece	of	work	...
"a great work"	0.3	0.5	...	0.1	0	0	...	0
"a great piece"	0.6	0.2	...	0.2	0	0	...	0
"a piece of work"	0.1	0.1	...	0.0	0	0	...	0
"piece of work"	0.9	0.3	...	0.5	0	0	...	0
"piece great"	1.0	0.6	...	0.4	1	0	...	0
"great film"	0.9	0.4	...	0.4	1	0	...	0
"film work"	0.7	0.9	...	0.1	1	0	...	0

("piece","work")

Bigrams carry context

Sentiment

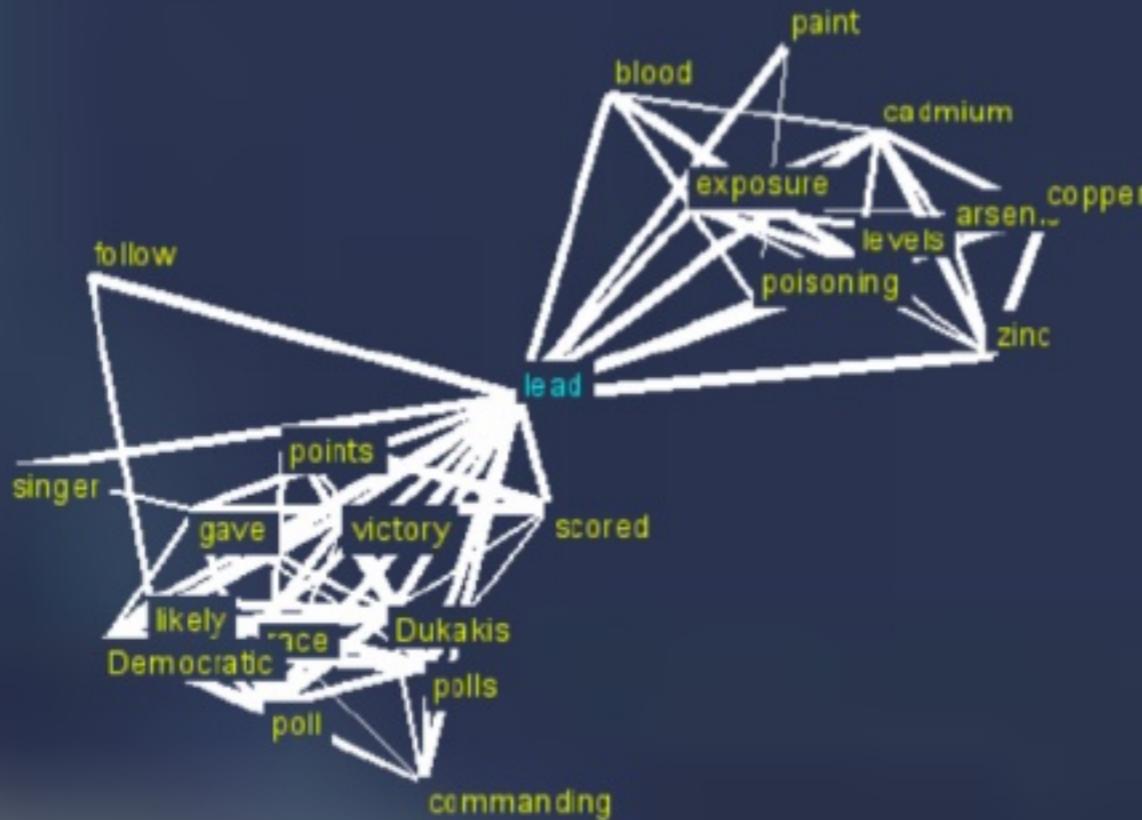
"a great work" Pos

"a great piece" Pos

"a piece of work" Neg

("piece", "work") is a bigram with negative sentiment

Find bigrams with GraphX



- If two words appear near each other in the training corpus, draw an edge
- (or increment edge count if there is already an edge)
- Edges with high count are the important bigrams to track

Diagram from Richter et al "Exploiting the Leipzig corpora collection"

Incorporating bigrams

	word2vec vectors (300 columns)				bigram presence indicators (300 columns)			
The	0.3	0.5	...	0.1	0	0	...	0
film	0.6	0.2	...	0.2	0	0	...	0
is	0.1	0.1	...	0.0	0	0	...	0
a	0.9	0.3	...	0.5	0	0	...	0
piece	1.0	0.6	...	0.4	1	0	...	0
of	0.9	0.4	...	0.4	1	0	...	0
work	0.7	0.9	...	0.1	1	0	...	0

("piece", "great", "film")
("piece", "work")

Idea

Bigrams carry context

Sentiment	
"a great work"	Pos
"a great piece"	Pos
"a piece of work"	Neg

("piece","work") is a bigram with negative sentiment



Diagram from Richter et al "Exploiting the Leipzig corpora collection"

Find bigrams with GraphX

- If two words appear near each other in the training corpus, draw an edge
- (or increment edge count if there is already an edge)
- Edges with high count are the important bigrams to track

Incorporating bigrams

	word2vec vectors (300 columns)				bigram presence indicators (300 columns)			
	The	film	is	a	piece	of	work	...
"a great work"	0.3	0.5	...	0.1	0	0	...	0
"a great piece"	0.6	0.2	...	0.2	0	0	...	0
"a piece of work"	0.1	0.1	...	0.0	0	0	...	0
"piece of work"	0.9	0.3	...	0.5	0	0	...	0
"piece great"	1.0	0.6	...	0.4	1	0	...	0
"great film"	0.9	0.4	...	0.4	1	0	...	0
"film work"	0.7	0.9	...	0.1	1	0	...	0

("piece","work")
("great","film")

Implementation

word2vec

CNN

Graph

stop words

Jupyter/Python

gensim



Keras

Adjacency
matrix



NLTK

Spark/Scala

DL4J
DEEPMLEARNING4J

DL4J
DEEPMLEARNING4J



NLTK

Prototype in Jupyter/Python, Implement in Spark/Scala

The screenshot shows a Jupyter Notebook window titled "sentiment20170420". The URL is "http://sicag175.us.oracle.com:8888/notebooks/sentiment20170420.ipynb". The kernel is set to Python 3. The notebook has five cells:

- In [9]:

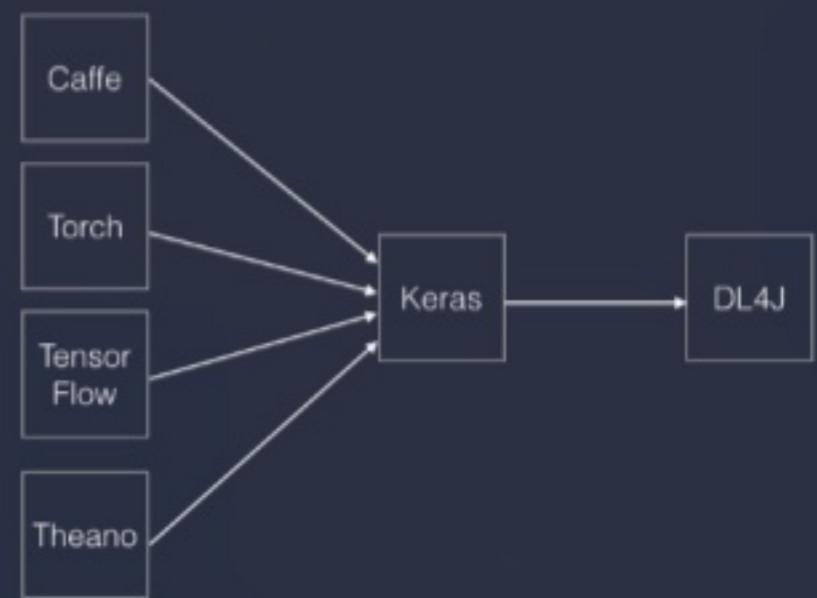
```
import keras
```

Using TensorFlow backend.
- In [10]:

```
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation
from keras.layers import Convolution1D, GlobalMaxPooling1D, Convolution2D, Global
```
- In [11]:

```
filter_length = 3
CNNDenseDropout = 0.2
dense_dims = 20
batch_size = 10
nb_epoch = 5
validation_split = 0.1
```
- In [12]:

```
# Reference standard usage of CNN/word2vec for sentiment analysis
# cf http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-patterned-after https://github.com/fchollet/keras/blob/master/examples/imdb_cnn
```



org.deeplearning4j.nn.modelimport.keras
Class KerasModellImport

DL4J can import Keras model (to predict in Spark), but not pipeline (to train & predict in Spark)

Implementation

word2vec

CNN

Graph

stop words

Jupyter/Python

gensim



Keras

Adjacency
matrix



NLTK

Spark/Scala

DL4J
DEEPMLEARNING4J

DL4J
DEEPMLEARNING4J



NLTK

Spark/Scala

DL4J
DEEPMLEARNING4J

word2vec on Spark technology choice



DL4J
DEEPMLEARNING4J

- SPARK-15328
- "Word2Vec import for original binary format"
- Unresolved
- PR #13725
- Doesn't work either

- Works



Neither distributes the model (1.5GB raw data)

DL4J

CNN on Spark technology choice

Spark Packages

mCNN
github.com/timveld/CNN

DL4J
DEEPMLEARNING4J

- Supports Spark, GPU
- Compared to Keras
- Develop code from test code

word2vec on Spark technology choice



DL4J
DEEPMLEARNING4J

- SPARK-15328
- "Word2Vec import for original binary format"
- Unresolved
- PR #13735
- Doesn't work either

- Works



Neither distributes the model (1.5GB raw data)

DL4J Word2Vec API Gotcha

readWord2VecModel

```
public static Word2Vec readWord2VecModel(java.lang.String path)
```

This method 1) Binary model, either compressed or not. Like well-known Google Model 2) Popular CSV word2vec text format 3) DL4j compressed format Please note: Only weights will be loaded by this method.

loadGoogleModelNonNormalized

@Deprecated

```
public static WordVectors loadGoogleModelNonNormalized(java.io.File modelFile,
                                                       boolean binary,
                                                       boolean lineBreaks)
                                                       throws java.io.IOException
```

Deprecated.

Loads the Google model without normalization being applied. PLEASE NOTE: Use this method only if you understand why you need not-normalized model. In all other cases please use `loadGoogleModel()` instead. Deprecation note: Please, consider using `readWord2VecModel()` or `loadStaticModel()` method instead

word2vec on Spark technology choice



DL4J
DEEPMLEARNING4J

- SPARK-15328
- "Word2Vec import for original binary format"
- Unresolved
- PR #13735
- Doesn't work either

- Works



Neither distributes the model (1.5GB raw data)

Implementation

word2vec

CNN

Graph

stop words

Jupyter/Python

gensim



Keras

Adjacency
matrix



NLTK

Spark/Scala

DL4J
DEEPMLEARNING4J

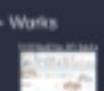
DL4J
DEEPMLEARNING4J



NLTK

DEEPMLEARNING4J

- SPARK-15328
- "Word2Vec Import for original binary format"
- Unresolved
- PR #13735
- Doesn't work either



Neither distributes the model (1.5GB raw data)

DL4J

DEEPMLEARNING4J

CNN on Spark technology choice

Spark Packages

mCNN

github.com/mhdyy/mCNN

- 0.5
- Pooling layers under development
- No check-ins for 2 years

DL4J

DEEPMLEARNING4J

- Supports Spark, GPU
- Compared to Keras:
 - Do your own minibatch split
 - Do your own batching
 - More parameters, more learning
 - Dependency on ND4J for BLAS
 - Has its own API, learning curve



Deep Learning
for Everyone

CNN on Spark technology choice

Spark Packages

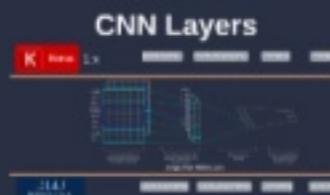
mCNN

github.com/hhbyyh/mCNN

- 0.5
- Pooling layers under development
- No check-ins for 2 years

DL4J DEEPMLEARNING4J

- Supports Spark, GPU
- Compared to Keras:
 - Do your own train/test split
 - Do your own batching
 - More parameters, more tweaking
- Dependency on ND4J for BLAS
 - Has its own API, learning curve



CNN Layers



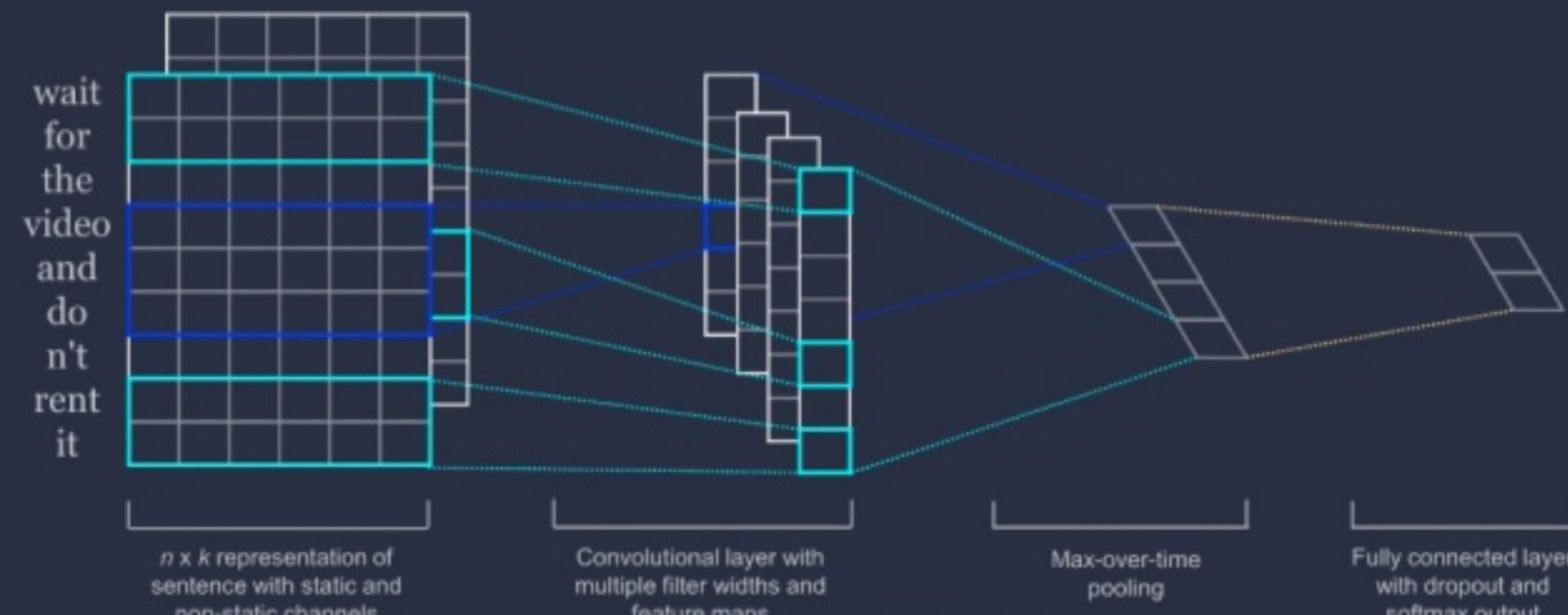
Keras 1.x

Convolution1D

GlobalMaxPooling1D

Dense(20)

Dense(1)



DL4J
DEEPMLEARNING4J

ConvolutionLayer

GlobalPoolingLayer

DenseLayer

OutputLayer

CNN on Spark technology choice

Spark Packages

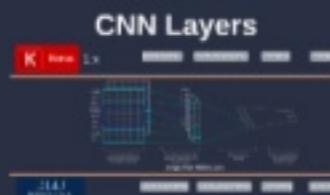
mCNN

github.com/hhbyyh/mCNN

- 0.5
- Pooling layers under development
- No check-ins for 2 years

DL4J DEEPMLEARNING4J

- Supports Spark, GPU
- Compared to Keras:
 - Do your own train/test split
 - Do your own batching
 - More parameters, more tweaking
- Dependency on ND4J for BLAS
 - Has its own API, learning curve



Implementation

word2vec

CNN

Graph

stop words

Jupyter/Python

gensim



Keras

Adjacency
matrix



NLTK

Spark/Scala

DL4J
DEEPMLEARNING4J

DL4J
DEEPMLEARNING4J



NLTK

Graphs in Python Technology Choice



scipy.sparse.csgraph

NetworkX



	1	2	3	4	5	6
1	0	1	1	0	0	0
2	1	0	0	1	0	0
3	1	0	0	1	0	0
4	0	1	1	0	1	0
5	0	0	0	1	0	1
6	0	0	0	0	1	0

Adjacency Matrix

Some Graph Philosophy

- Graphs are isomorphic to matrices
- Efficient if a lot of "missing" edges
- Edge weights escape binary edge/no-edge

Implementation

word2vec

CNN

Graph

stop words

Jupyter/Python

gensim



Keras

Adjacency
matrix



NLTK

Spark/Scala

DL4J
DEEPMLEARNING4J

DL4J
DEEPMLEARNING4J

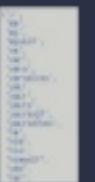


NLTK

Graph stop words

Usage of Stop Words

- From NLTK, just read directly from `nlk_data/stopwords.zip/stopwords/english`
- 153 words
- Used for bigram detection (co-occurrence graph) but not for stacking the word2vec vectors for the CNN input matrix



Usage of Stop Words

- From NLTK, just read directly from `nltk_data/stopwords.zip/stopwords/english`
- 153 words
- Used for bigram detection (co-occurrence graph) but not for stacking the word2vec vectors for the CNN input matrix

```
['i',  
 'me',  
 'my',  
 'myself',  
 'we',  
 'our',  
 'ours',  
 'ourselves',  
 'you',  
 'your',  
 'yours',  
 'yourself',  
 'yourselves',  
 'he',  
 'him',  
 'his',  
 'himself',  
 'she',  
 'her',
```

Implementation

word2vec

CNN

Graph

stop words

Jupyter/Python

gensim



Keras

Adjacency
matrix



NLTK

Spark/Scala

DL4J
DEEPMLEARNING4J

DL4J
DEEPMLEARNING4J



NLTK

Results

Data Used

Data Source

- Cornell movie review data
 - <http://www.cs.cornell.edu/people/pabo/movie-review-data>
 - 5,331 positive reviews, 5,331 negative reviews
 - Each 2 to 51 words long

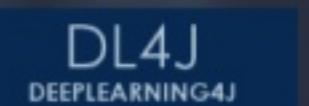
the rock is destined to be the 21st century's new "coran" and that he's going to make a splash even greater than Arnold Schwarzenegger... just don't say domino or stonehenge.
the gorgeously ridiculous continuation of "the kid of the ring" trilogy is so huge that a column of words cannot adequately describe it other than to say it's... well... huge.
you sometimes like to go to the movies to have fun... yassassin is a good place to start.

Data Prep

```
def readTweets(filename:String, isPositive:Boolean) = sc.makeRDD(  
    Source.fromFile(filename)(Codec.UTF8.decoder.onMalformedInput(CodingErrorAction.IGNORE))  
    .getLines().toList.map(x => ("[" + x + "]").r.replaceAllIn(x, "").split(" ")), isPositive))
```

- Strip non-alpha
 - Did not same-case

Accuracy



<i>CNN/word2vec</i>	76%	68%
<i>CNN/word2vec + co-occurrence graph</i>	80%	70%

Data Used

Data Source

- Cornell movie review data
- <http://www.cs.cornell.edu/people/pabo/movie-review-data/>
- 5,331 positive reviews, 5,331 negative reviews
- Each 2 to 51 words long

the rock is destined to be the 21st century's new " conan " and that he's going to make a splash even greater than arnold schwarzenegger , jean-claud van damme or steven segal .
the gorgeously elaborate continuation of " the lord of the rings " trilogy is so huge that a column of words cannot adequately describe co-writer/director peter jackson's expanded vision of j . r . r . tolkien's middle-earth .
effective but too-tepid biopic
if you sometimes like to go to the movies to have fun , wasabi is a good place to start .

Data Prep

```
def readTweets(filename:String, isPositive:Boolean) = sc.makeRDD(  
    Source.fromFile(filename)(Codec.UTF8.decoder.onMalformedInput(CodingErrorAction.IGNORE))  
        .getLines.toList.map(x => ("[^a-zA-Z ]".r.replaceAllIn(x,"")).split(" "), isPositive)))
```

- Strip non-alpha
- Did not same-case

Accuracy



CNN/word2vec 76% 68%

*CNN/word2vec +
co-occurrence graph* 80% 70%

CNN Plus Co-Occurrence Graphs

Idea

Bigrams carry context	
	Sentiment
"big giant nouns"	Poss.
"big giant pieces"	Poss.
"a piece of work"	Neg.

(Please) work! It's a slogan with negative sentiment

**Find bigrams
with GraphX**

Incorporating bigrams

Implementation

The diagram illustrates the relationships between various NLP frameworks. At the top center is the **gensim** library. To its left, under the heading **Jupyter/Python**, are **word2vec**, **CNN**, **Graph**, and **stop words**. To its right, under the heading **Spark/Scala**, are **Keras** and **Graphviz**. Below **Keras** is the **Adjacency matrix**. At the bottom center is the **NLTK** library.

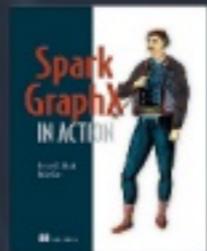
Results

Data Used

Accuracy

Dead Ends

Bad idea #1: 2D CNN



Bad idea #2: Semi-Supervised Learning of good/bad words



Bad idea #3: Co-occurrence graph to identify unigrams

- Really just bag of words

Bad idea #1: 2D CNN

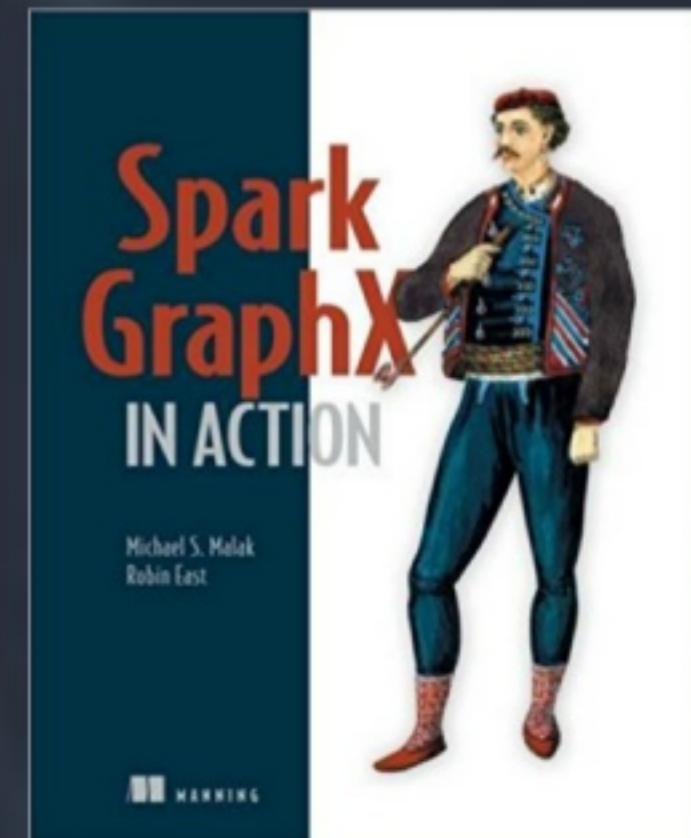
wait
for
the
video
and
do
n't
rent
it



Reorder columns so that similar columns are near each other



Column order determined by
minimum spanning tree of
word2vec columns



Bad idea #2: Semi-Supervised Learning of good/bad words

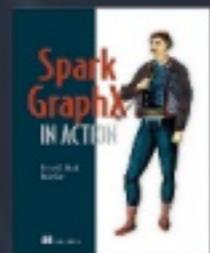


Bad idea #3: Co-occurrence graph to identify unigrams

- Really just bag of words

Dead Ends

Bad idea #1: 2D CNN



Bad idea #2: Semi-Supervised Learning of good/bad words



Bad idea #3: Co-occurrence graph to identify unigrams

- Really just bag of words

Sentiment Analysis

Prior Techniques



CNN Plus Co-Occurrence Graphs

Idea

Implementation

Results

Bad idea #1: 2D CNN



Bad idea #2: Semi-Supervised Learning of good/bad words



Bad idea #3: Co-occurrence graph to identify unigrams

=> Just bag of words

Dead Ends

Neuro-Symbolic Sentiment Analysis



Michael Malak

June 7, 2017

ORACLE® Cloud



Big Data Preparation



Analytics Cloud

Summary

- Classic/symbolic AI and neuro-symbolic just now out of winter
- OK to say "AI" now
- Can improve over neural-only techniques
- Keras/Jupyter great prototyping tool
- DL4J great for word2vec and CNN under Spark (and GPU)
- Source code: technicaltidbit.com in 2-4 weeks

Neuro-Symbolic Sentiment Analysis



Michael Malak

June 7, 2017

ORACLE® Cloud



Big Data Preparation



Analytics Cloud