



Quora Question Pairs Competition

YesofCourse Team Solution

Pang Liang, Fan Yixing, Hou Jianpeng,
Yue Xinyu, Niu Guocheng



中国科学院网络数据科学与技术重点实验室
Key Laboratory of Network Data Science & Technology, CAS

Content

- Introduction
- Solution
 - Pre-processing
 - Feature-Engineering
 - Deep Model
 - Traditional Model
 - Stacking
 - Post-processing
- Conclusion

Team Introduction



Niu Guocheng

Master
@Baidu

Researches: Natural
Language Processing and
Machine Learning



Pang Liang

PhD. candidate
@ICT

Researches: Information
Retrieval, Matching
Learning, Deep learning



Hou Jianpeng

Master
@Google

Researches: Machine
Learning and Distributed
Computing



Fan Yixing

PhD. candidate
@ICT

Researches: Information
Retrieval, Matching
Learning, Deep learning



Yue Xinyu

Master
@ICT

Researches:Recommendat
ion, Data Mining, Machine
Learning

Team Introduction

1st Place, Awarded in **2016 BYTECUP**

1st Place, Awarded in **China Telecom Big Data Application Contest**

1st Place, Awarded in **SIGHAN-2015 Chinese Spelling Check Task**

1st Place, Awarded in **RecSys2013: Yelp Business Rating Prediction**

Task Problem

Quora

A place to share knowledge and better understand the world



“ Why is WeChat more popular than
WhatsApp in China given that
WhatsApp is more popular elsewhere? ”



“ WeChat is quite popular in China. Are
there many people from other countries
using WeChat to contact each other? ”

Task Dataset

- Dataset Statistics

!!! Computer-generated pairs

	Question pairs	Distinct pairs	Total questions	Pos vs Neg
Training	404,290	404,258	537,373	36.92%
Testing	2,345,796	2,339,396	4,340,277	16.5%

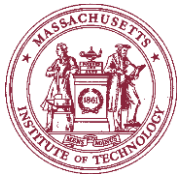
- Dataset Statistics

id	question1	question2	Is_duplicate
0	How do I prevent breast cancer?	Is breast cancer preventable?	0
1	How does 3D printing work?	How do 3D printing work?	1

- Evaluation

$$\text{logloss} = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^M y_{i,j} \log(p_{i,j})$$

Task Participants



3307

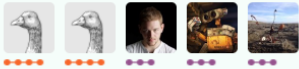


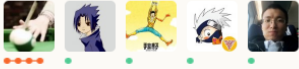

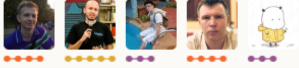

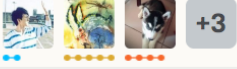

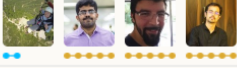


Kaggle Rankings

1			Gilberto Titericz Junior
2			Stanislav Semenov
3			Μαριος Μιχαηλιδης KazAnova
4			Faron
5			idle_speculation
6			Eureka
7			Silogram
8			Little Boat
9			utility
10			raddar

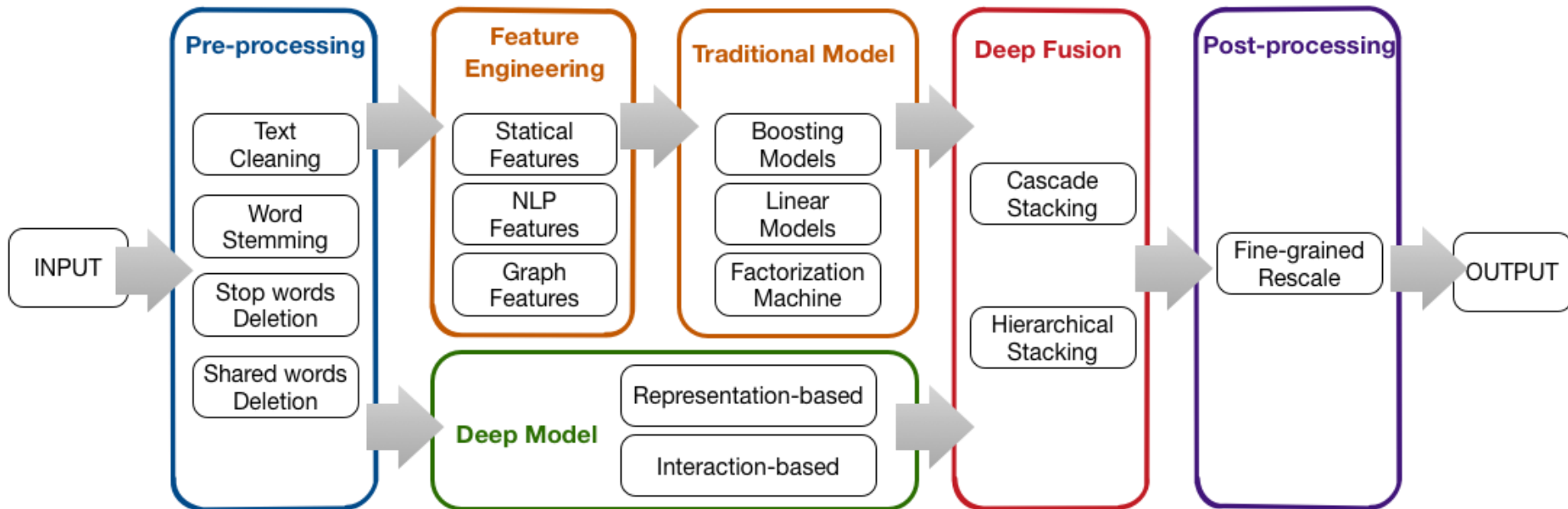


Final Leaderboard

#	△pub	Team Name	Kernel	Team Members	Score ?	Entries
1	—	DL guys			0.11580	263
2	—	Depp Learning			0.11670	196
3	—	Jared Turkewitz & sjv			0.11756	178
4	—	YesOfCourse			0.11768	189
5	—	Qingchen KazAnova Faron			0.11851	219
6	—	LAMAA power			0.11887	406
7	▲2	aphex34			0.12072	166
8	—	NLPFakers			0.12239	250
9	▼2	Unduplicated Duplicates			0.12248	314
10	▲1	🎵 b.a.s.s. 🎵			0.12296	271

Architecture

The framework of our solution:



Content

- Introduction
- Solution
 - **Pre-processing**
 - Feature-Engineering
 - Deep Model
 - Traditional Model
 - Stacking
 - Post-processing
- Conclusion

Preprocessing

Made some different versions of original data:

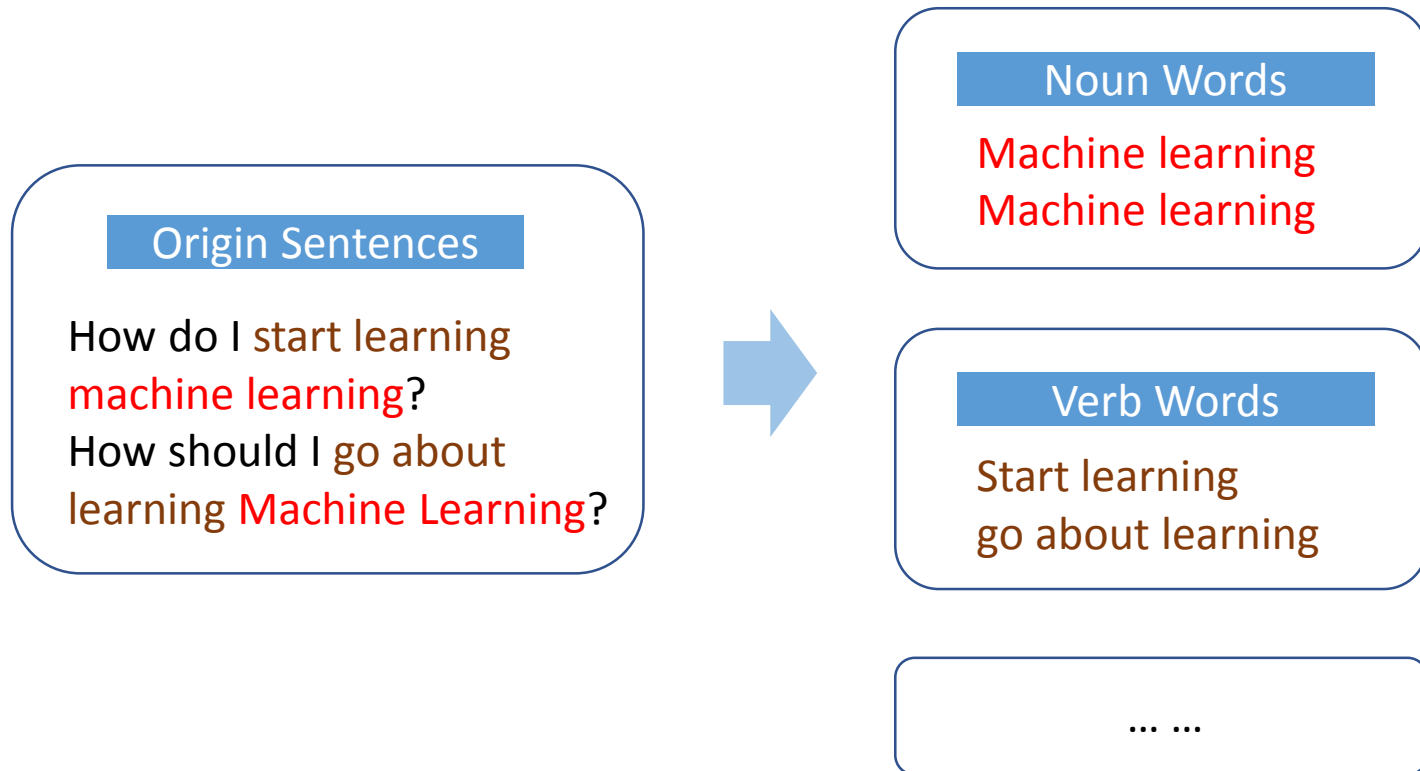
- Lower case
- Text cleaning
- Stop words removal
- Word stemming
- Punctuation Cleaning
- Shared words deletion
- Part of Speech Group

Preprocessing • Text Cleaning

- Substitute Abbreviation
 - ' n't ' => ' not '
 - ' that's ' => ' that is '
 - ' US ' => ' America '
- Substitute Special Character
 - ' \\$ ' => ' dollar '
 - ' ₹ ' => ' rs '
- Substitute Numbers
 - ' one ' => ' 1 '
 - ' 6k ' => ' 6000 '

Preprocessing • POS Group

- Extract different part of speech from origin sentences.



Content

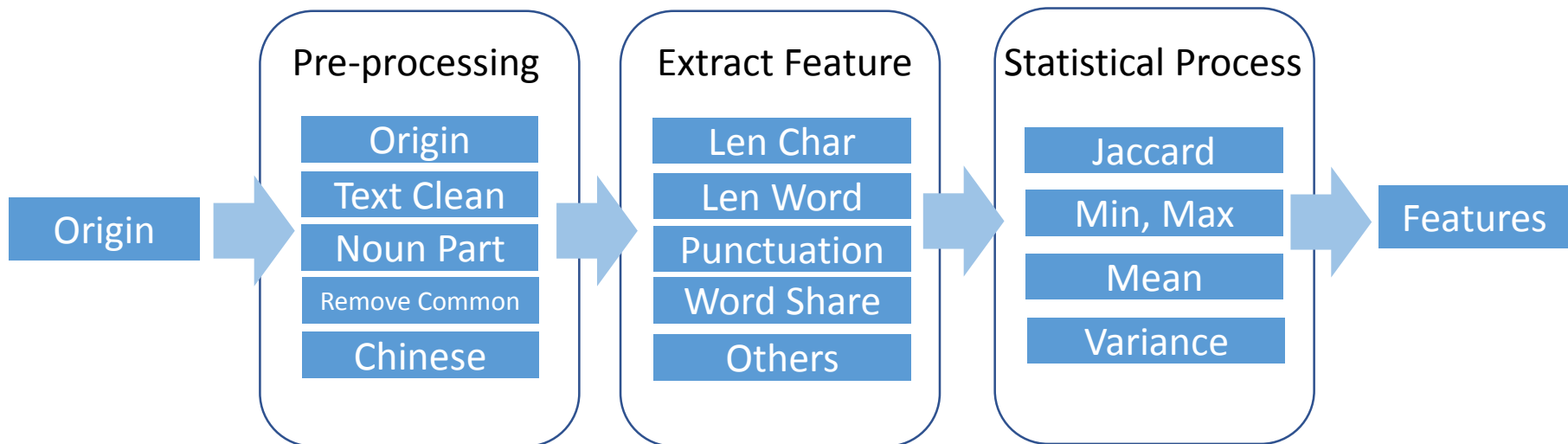
- Introduction
- Solution
 - Pre-processing
 - **Feature-Engineering**
 - Deep Model
 - Traditional Model
 - Stacking
 - Post-processing
- Conclusion

Feature Engineering

- Statistical
 - Powerful Words
 - Interrogative Words
 - FuzzyWuzzy
- NLP
 - Topic Model
 - Key Phrase Extraction
 - Dependency Parsing
 - Differential Analysis
 - Cross-Language Features
- Representation
 - Word Level
 - Sentence Level
- Graph
 - Nodes and Edges
 - Structure Information
 - Propagation Information

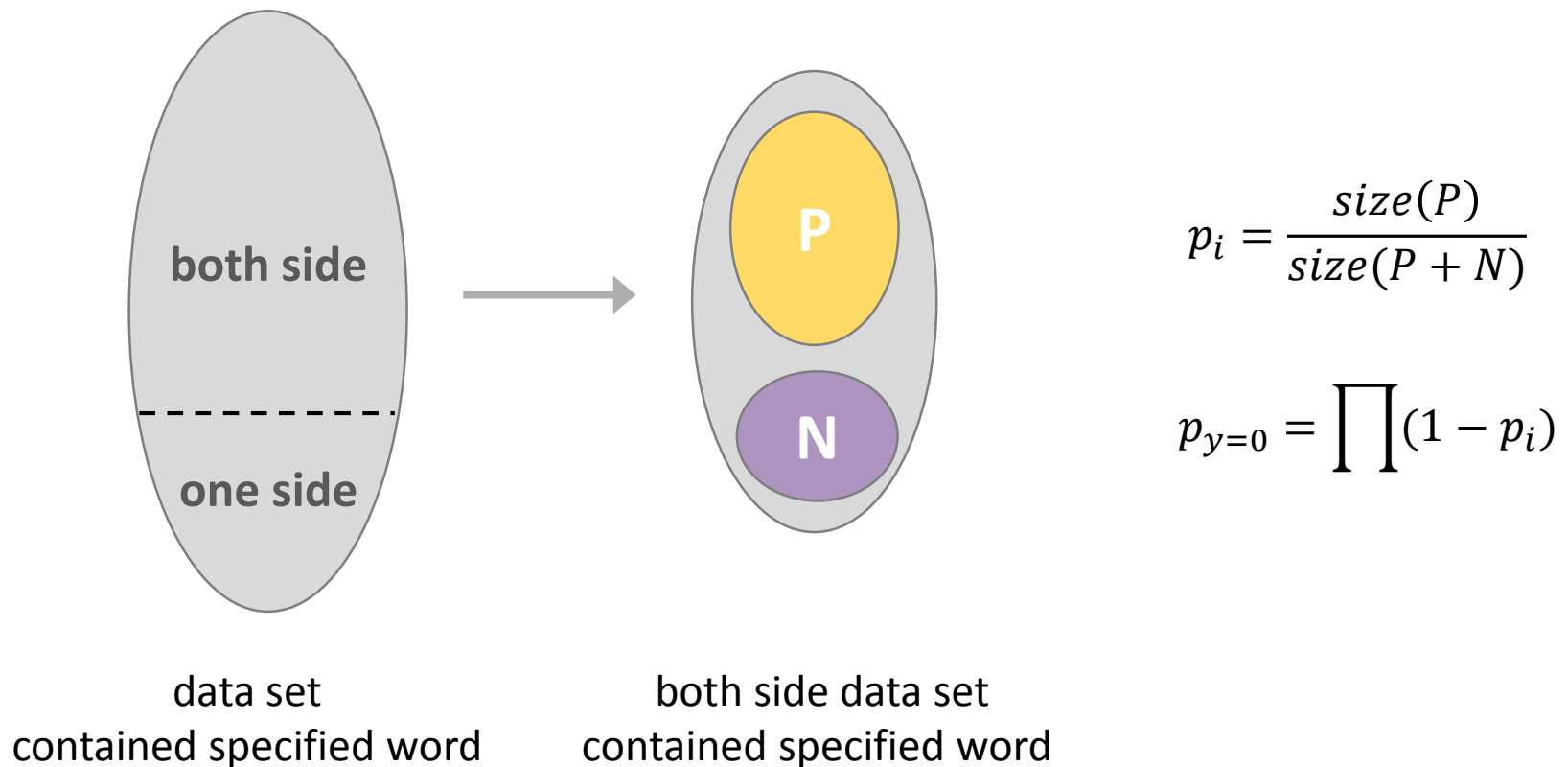
Statistical Features

- Len Char, Len Word, Len Vowel, Punctuation Num, Word Match Share et al
- Multi Channel including Origin, Text Cleaning
- Multi Process Method including Jaccard Ratio, Min, Max, Mean, Variance et al



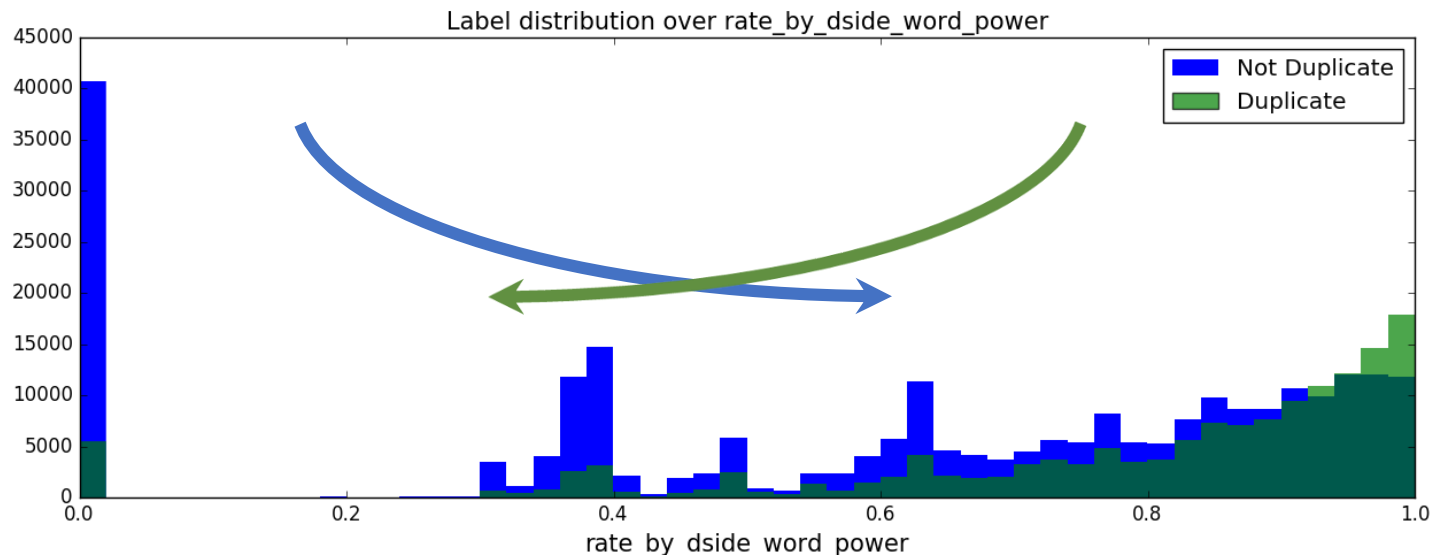
Statistical Features • Powerful Words

- Words which make two sentences express same meaning.



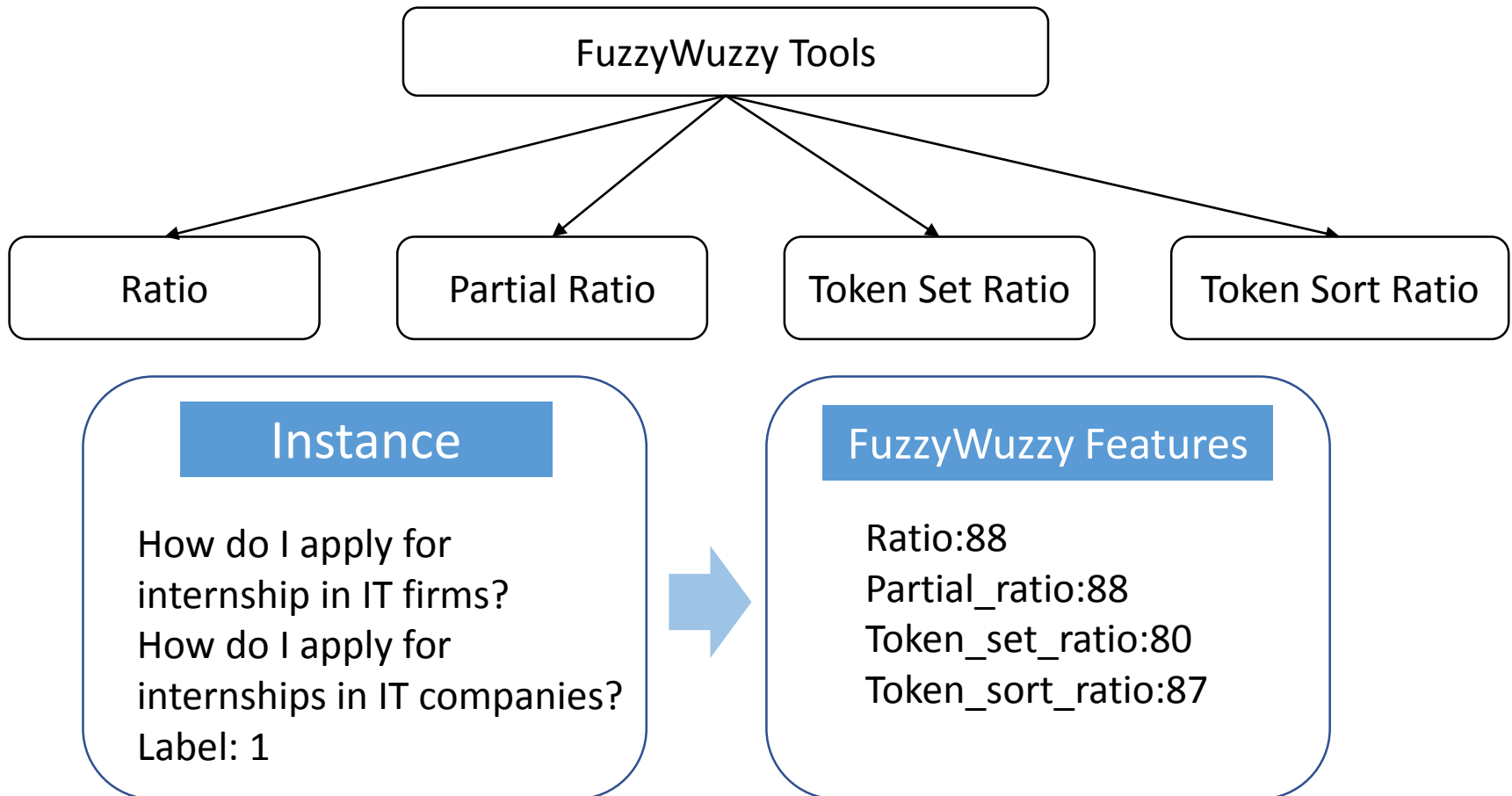
Statistical Features • Powerful Words

- Words which make two sentences express same meaning.



Statistical Features • FuzzyWuzzy

- Edit Distance calculation
- Multi types edit distance catch various info

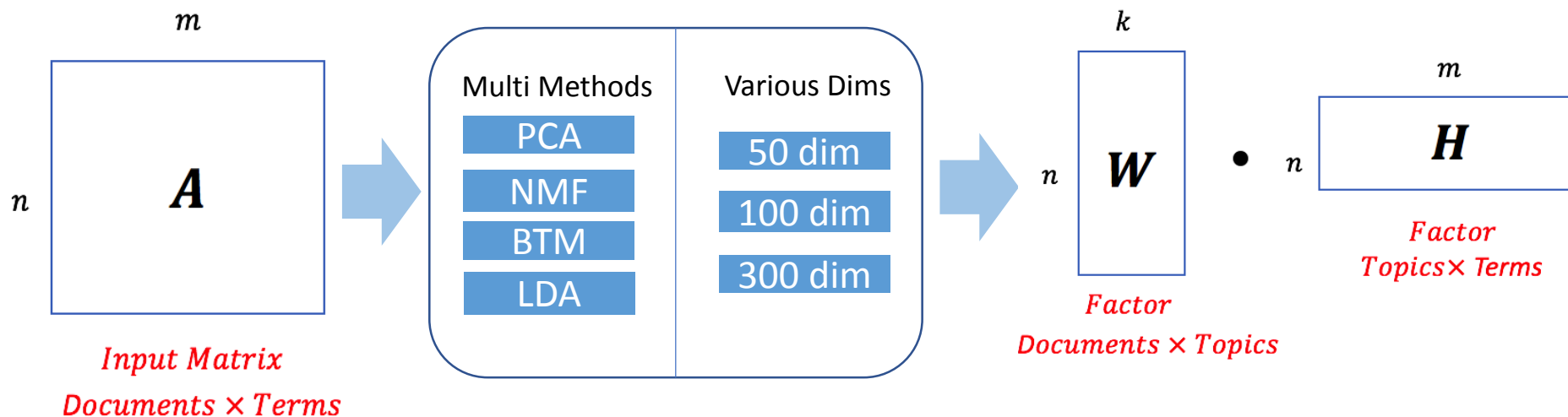


NLP Features

- Topic Model
 - PCA、 NMF、 BTM、 LDA
- Key Phrase Extraction
 - SG Rank
- Dependency Parsing
 - Principal component extraction
 - Semantic tree
- Differential Analysis
 - Part-of-speech、 Named-entity、 Brown-cluster
- Cross-Language Features
 - Translate into Chinese

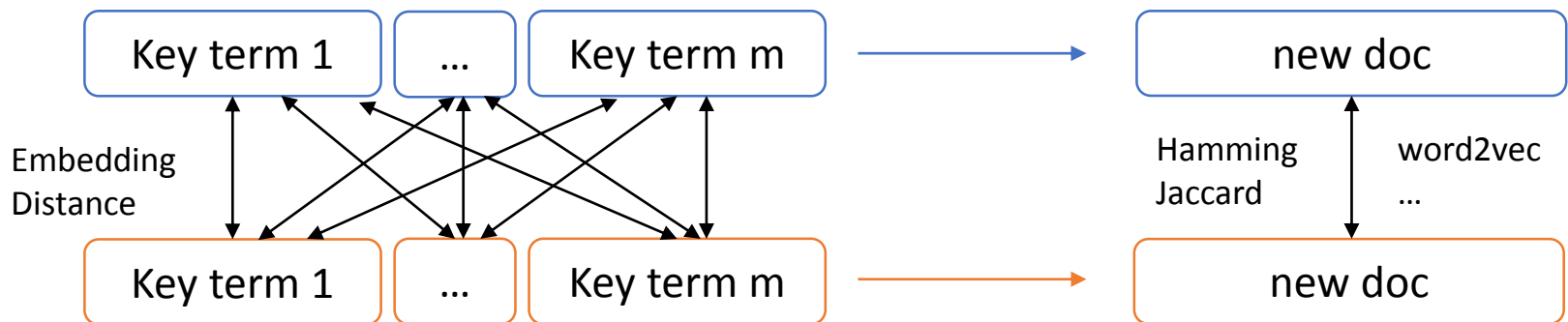
Topic Model Features

- Construct TF-IDF Matrix
- Matrix Factorization including PCA、NMF、BTM、LDA et al with several dimensions



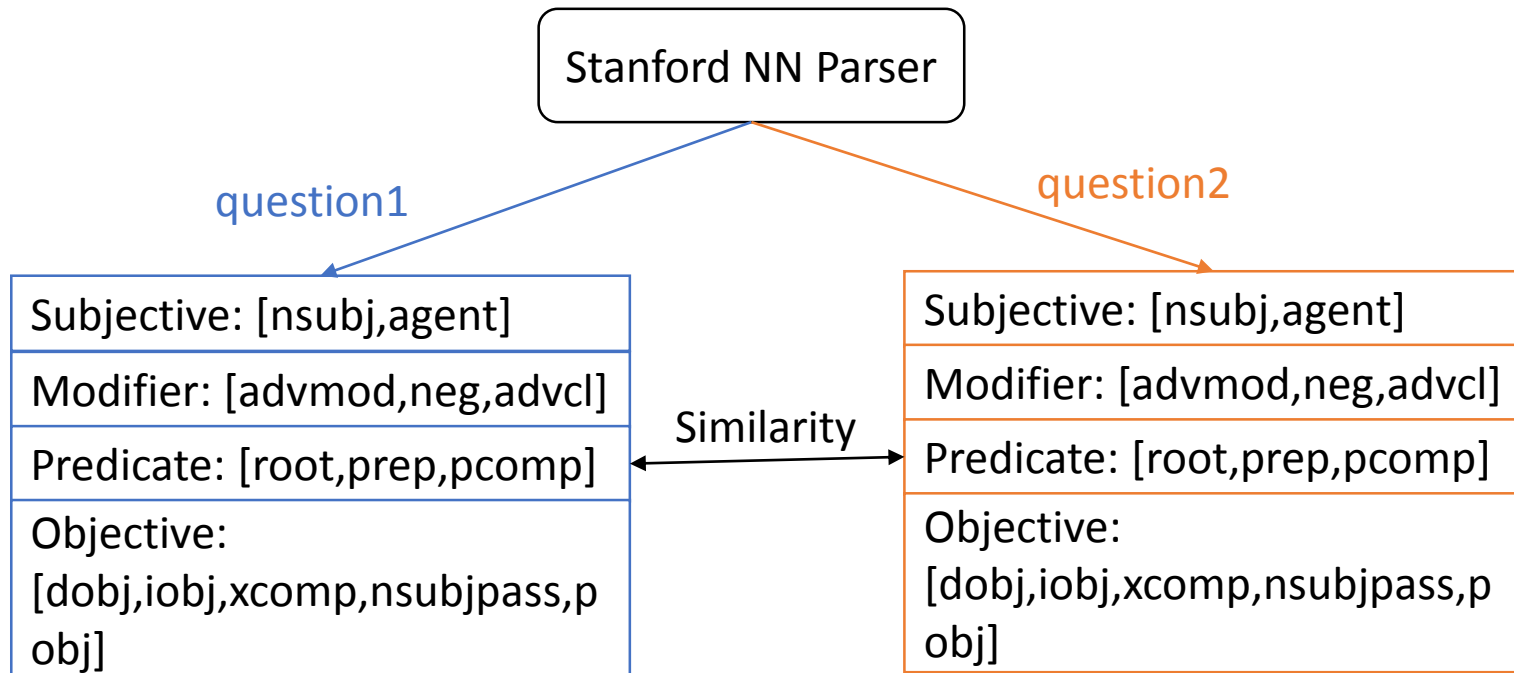
Key Phrase Extraction

- Unsupervised Key-phrase Extraction
- A hybrid statistical-graphical algorithm SGRank[Sem 2015]
- Reattach key terms as the new doc instead of origin one



Dependency Parsing

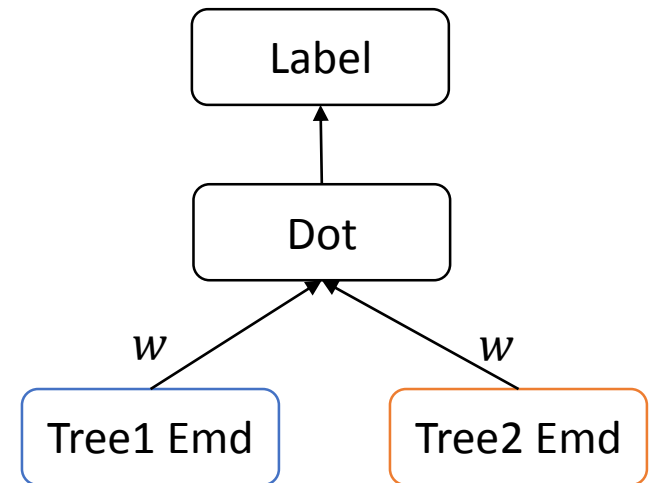
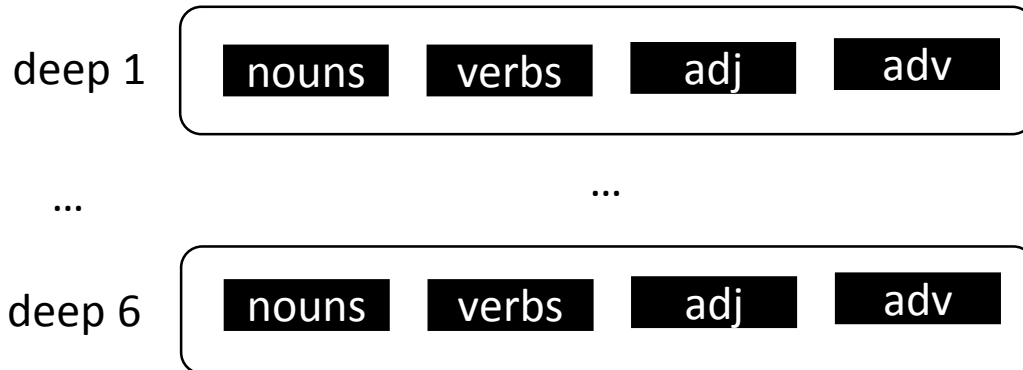
- Sentence principal component extraction
- Light rules can be summed up based on Dep-Parser



Dependency Parsing

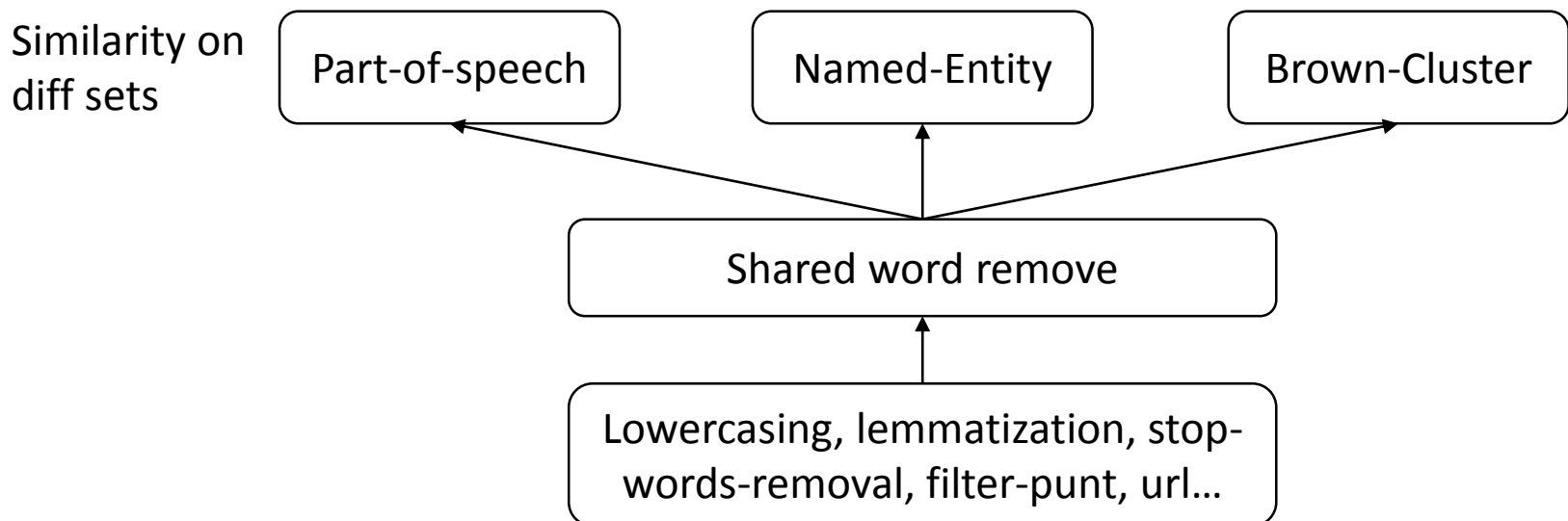
- Semantic information can put to use on trees
- Weights of part-of-speech on different depth are different
- Weights can be supervised learned on training set

$$TreeEmd = \sum_{i=1}^6 \sum_{j=1}^4 PosEmd_{i,j} * w_{i,j}$$



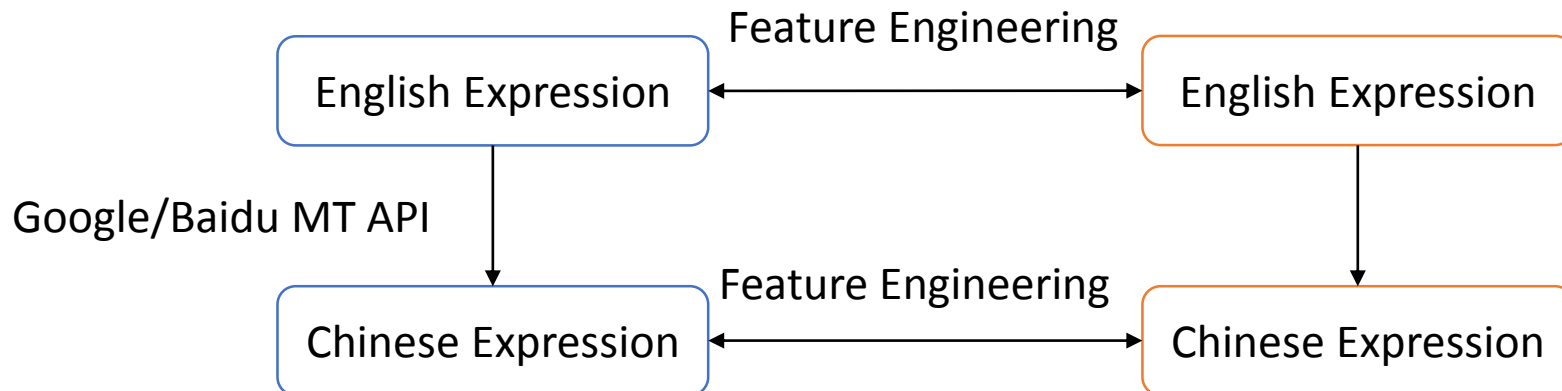
Differential Analysis

- Same words between pairs can confuse us in some ways
- Otherness is a nice point of view to noise elimination
- Pos, Named-Entity, Word-Cluster features based on diff sets



Cross-Language(Brain Hole)

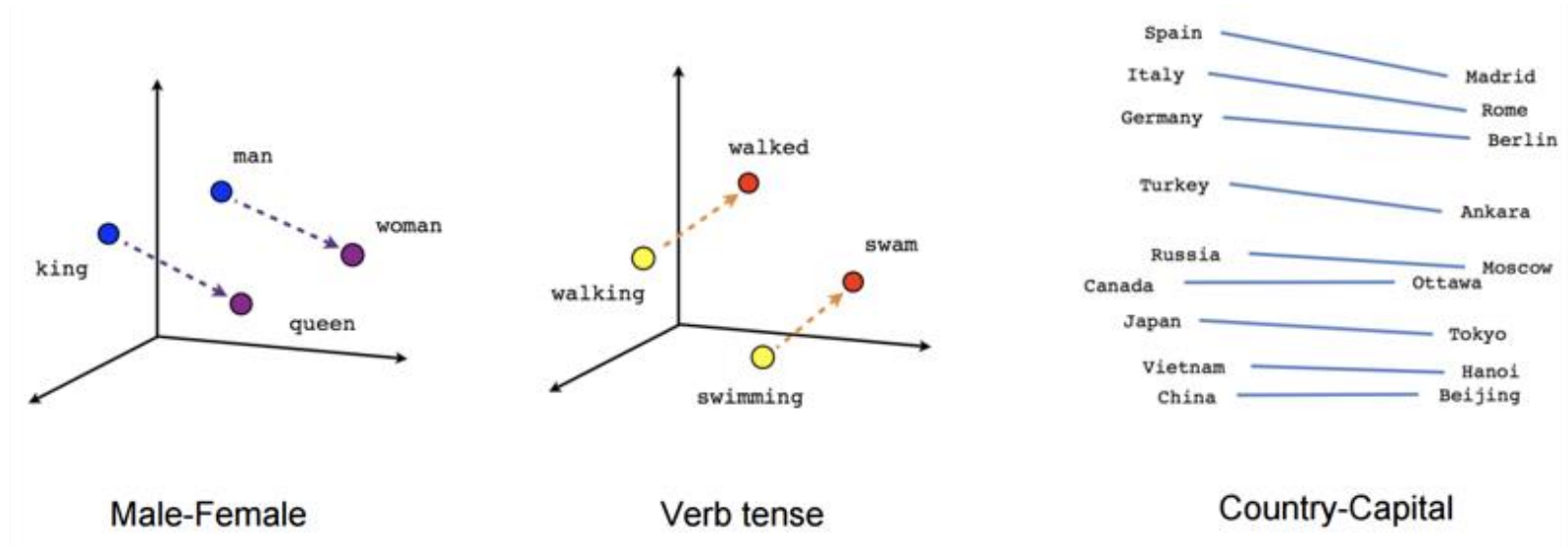
- Translate English to Chinese or other languages
- Same feature engineering repeat on translated languages
- Machine Translation are better on Polysemy and Unification



Representation Features

- Word Embedding
 - Latent Semantic Indexing (LSI)
 - Word2vec
 - Glove
- Sentence Embedding
 - Weighted Sum of Word Embedding
 - Paragraph Vector
 - Skip-Thought Vector

Word Representation



$$\text{vector}[\text{Queen}] = \text{vector}[\text{King}] - \text{vector}[\text{Man}] + \text{vector}[\text{Woman}]$$

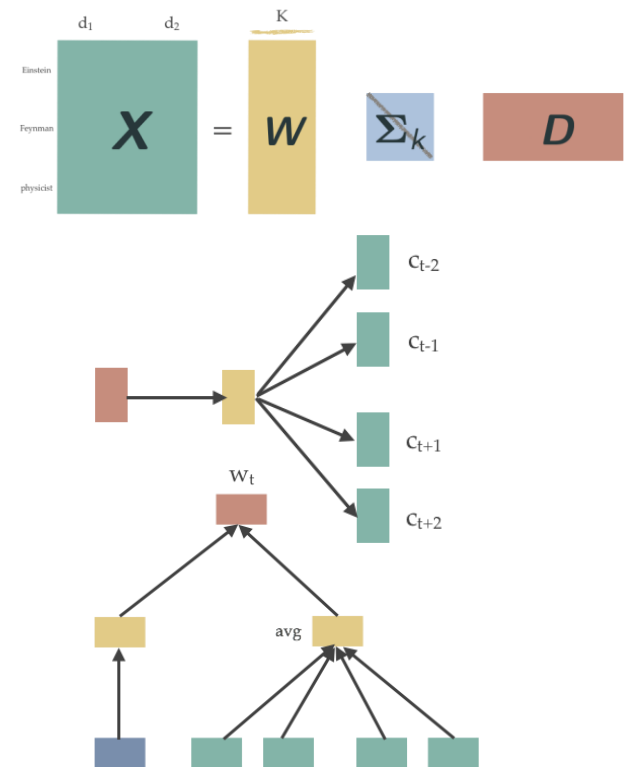
- Semantically similar words are mapped to nearby points
- Efficient representation: continuous vector space

Word Representation

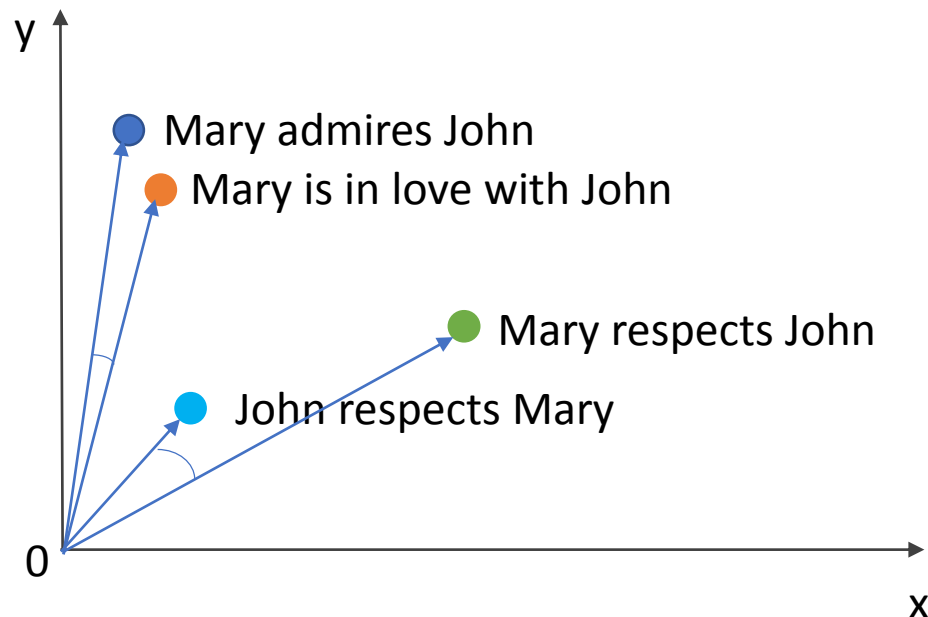
Context is the key in distributional hypothesis.

What type of context you use decides what kind of meaning or semantic relations between words you obtain.

- LSI: SVD decomposition of **Word-Document** Co-occurrence Matrix
- Word2vec: word predict **local context** words.
- HDC: word predict **local context** & **global document** wide



Sentence Representation

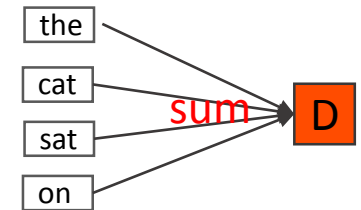


- Semantically similar sentence are mapped to nearby points
- Fluent & efficient representation:
 - Continuous vector space
 - Map varied-length sentence to fix-length vectors

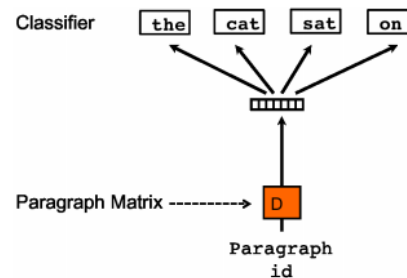
Sentence Representation

Internal word info. & **external** sentence info

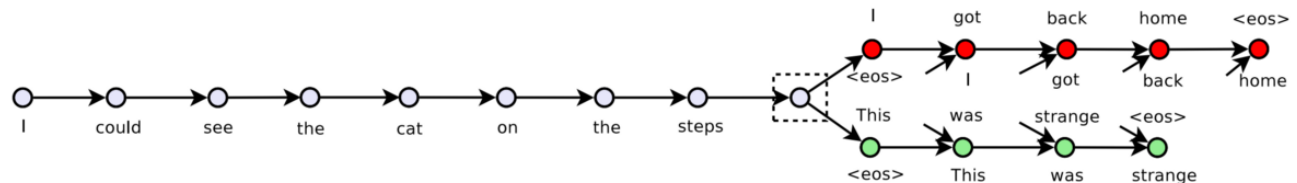
- Weighted Sum of Word Embeddings



- Paragraph Vector



- Skip Thought



Representation Features

- Word vector features

- Max matching signals
- Word move distance
- noun phrases similarities

- Sentence vector features

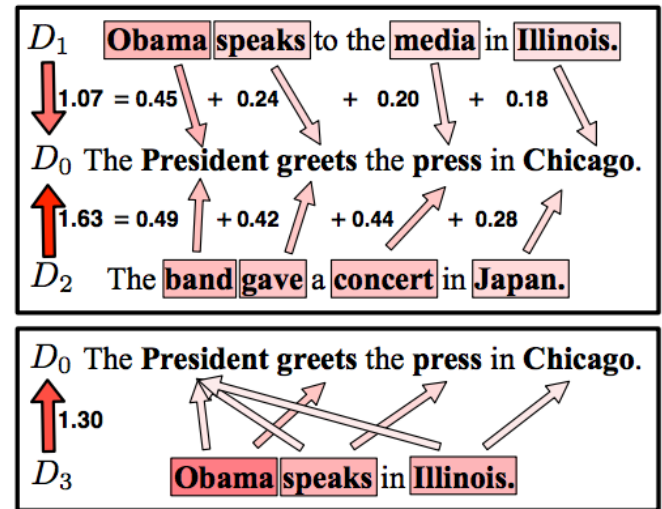
- Cosine Similarity $\cos(\theta) = \frac{x \cdot y}{|x||y|}$

- canberra $d = \sum_{k=1}^n \frac{|x_k - y_k|}{x_k + y_k}$

- rmse_distance $d = \sqrt{\frac{\sum_{k=1}^n (x_k - y_k)^2}{n}}$

- Minkowski (l = 1) : City Block/Manhattan $d = \sum_{k=1}^n |x_k - y_k|$

- Minkowski (l = 2) : Euclidean $d = \sqrt{(x - y)(x - y)^T}$



Graph

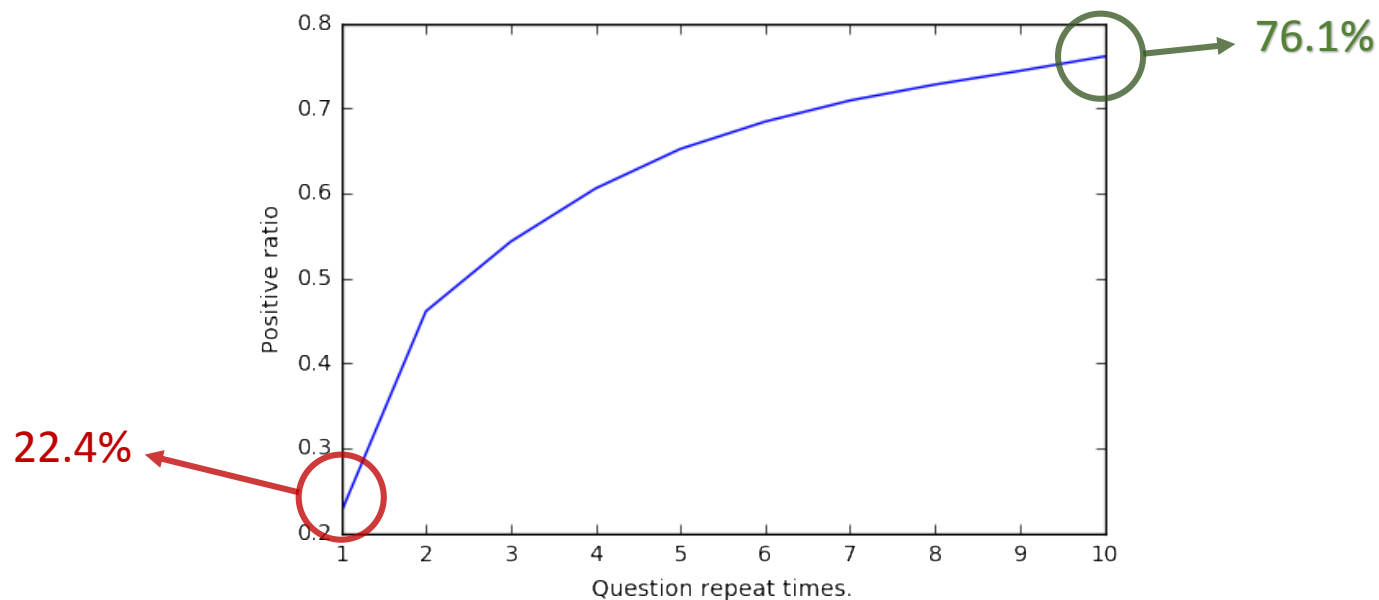
- Build Graph
 - Directed Graph / Undirected Graph
 - Co-occurrence Based Graph / Similarity Based Graph
- Statistic of nodes and edges
 - In-degree of the nodes
- Structure information
 - Component analysis
 - Clique analysis
- Propagation information
 - PageRank / Hits
 - Neighbor analysis
 - Features propagation (shortest path)

Build Graph

- Co-occurrence Based Graph
 - Node: Question
 - Edge: Question Pair in Train/Test dataset
 - Represent the **linking** properties of the questions.
- Similarity Based Graph
 - Node: Question
 - Edge: Similar Question Pairs evaluate by IR (BM25)
 - Represent the **similarity** properties of the questions.

Statistic of Nodes and Edges

- In-degree of the nodes
 - In Co-occurrence Based Graph, it is the same as the **repeat times** of each question in Train/Test set.

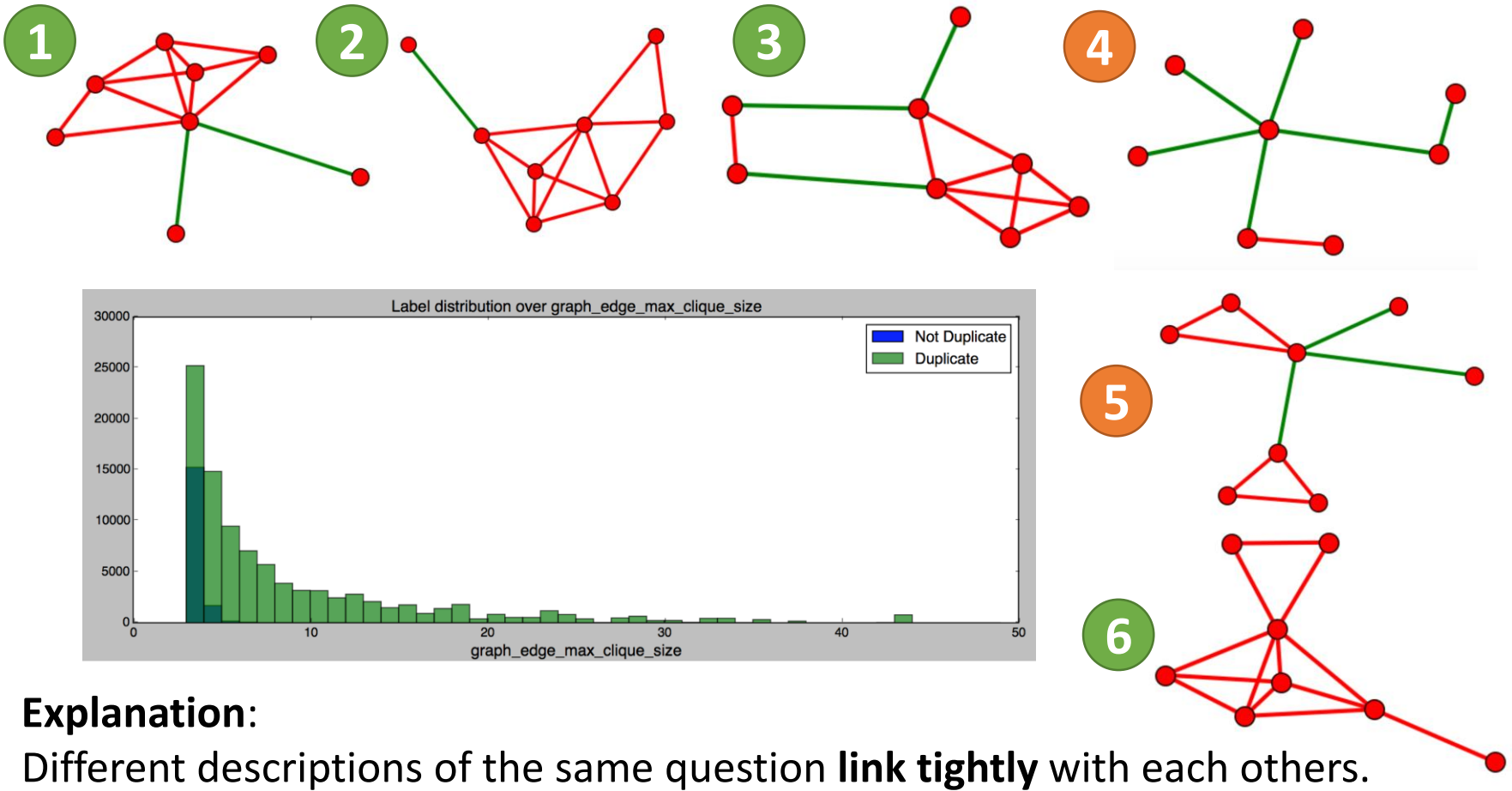


Explanation:

Frequent viewed questions = Hot topics = High duplicate ratio

Structure information

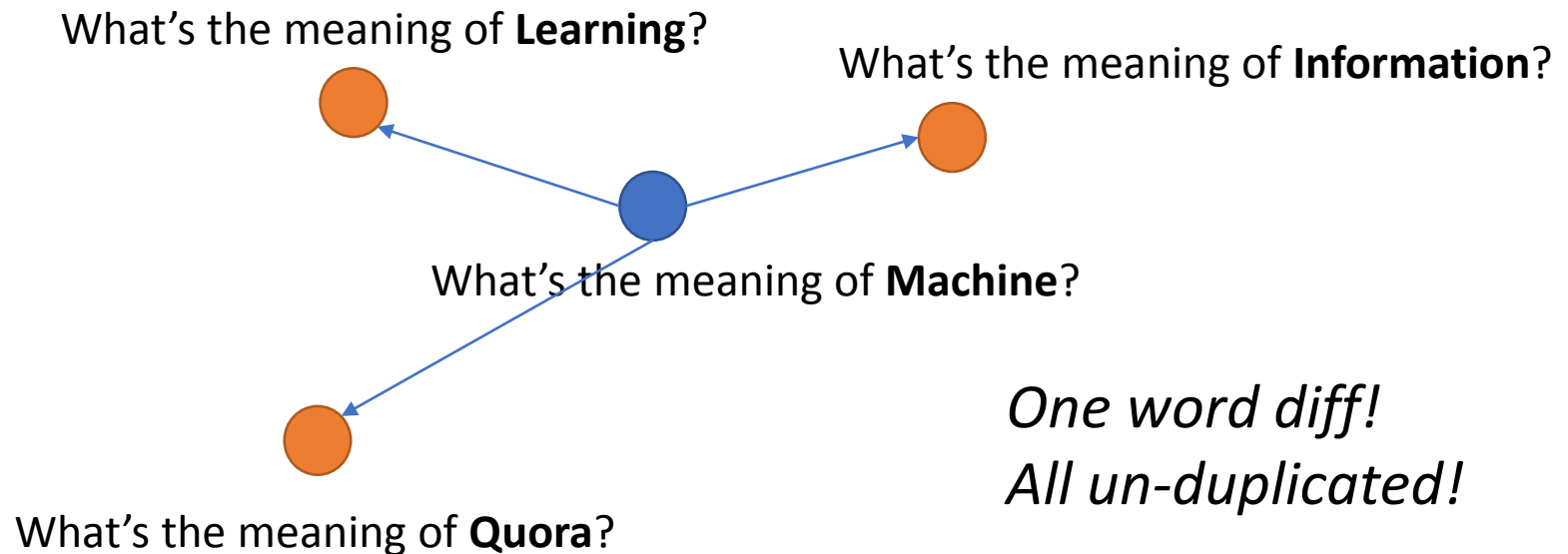
- Component & Clique analysis



Propagation Information

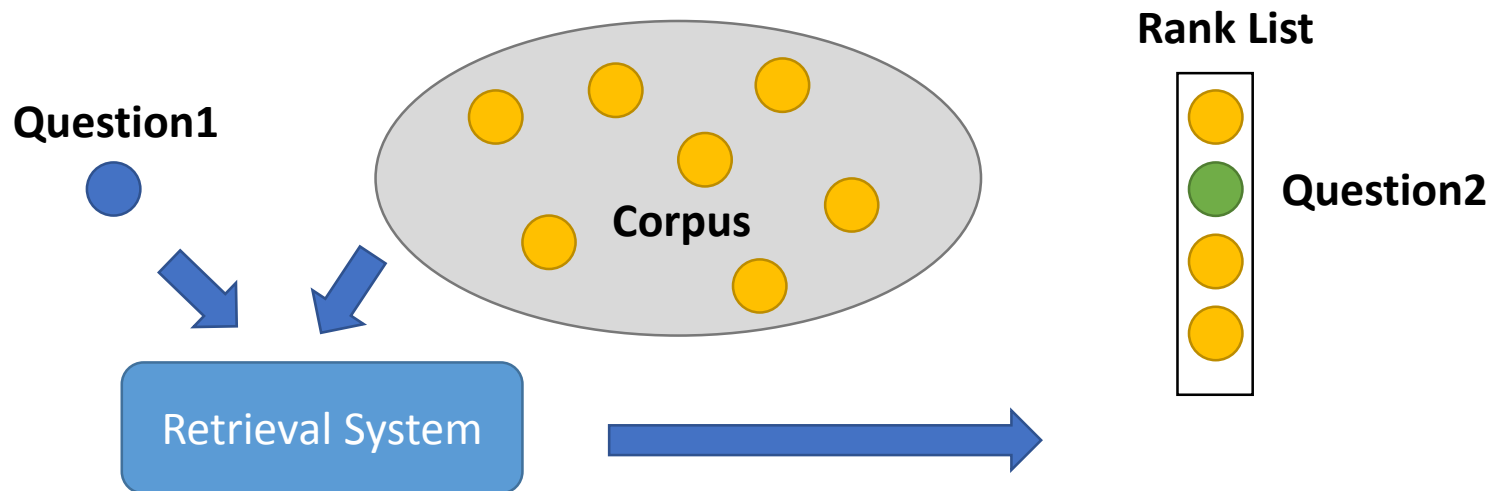
- Neighbor Analysis

- In Co-occurrence Based Graph, neighbors' features can be used to represent current question.
- Such as word shared count



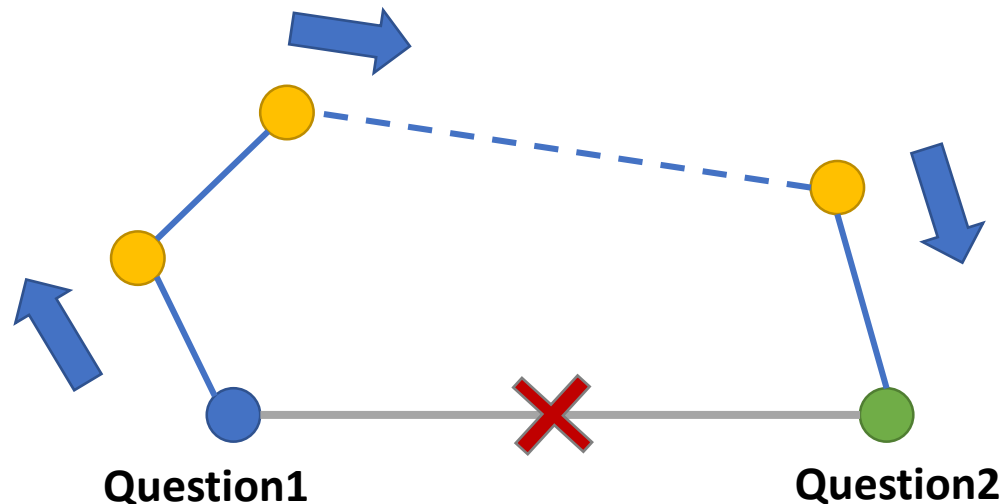
Propagation Information

- Neighbor Analysis
 - In Similarity Based Graph, the rank information of the neighbors are helpful.
 - We can treat the whole question set as a **corpus**, and the Quora System **retrieval/recommend** similar questions from corpus to users.



Propagation Information

- Feature Propagation
 - Put the feature value on the edge of the graph (Co-occurrence Based Graph).
 - Then calculate the shortest path value on the graph with removing current edge.




Content

- Introduction
- Solution
 - Pre-processing
 - Feature-Engineering
 - **Deep Model**
 - Traditional Model
 - Stacking
 - Post-processing
- Conclusion

Deep Models

- The text matching problem, in general, can be formalized as

$$\text{Match}(T_1, T_2) = F(\phi(T_1), \phi(T_2))$$



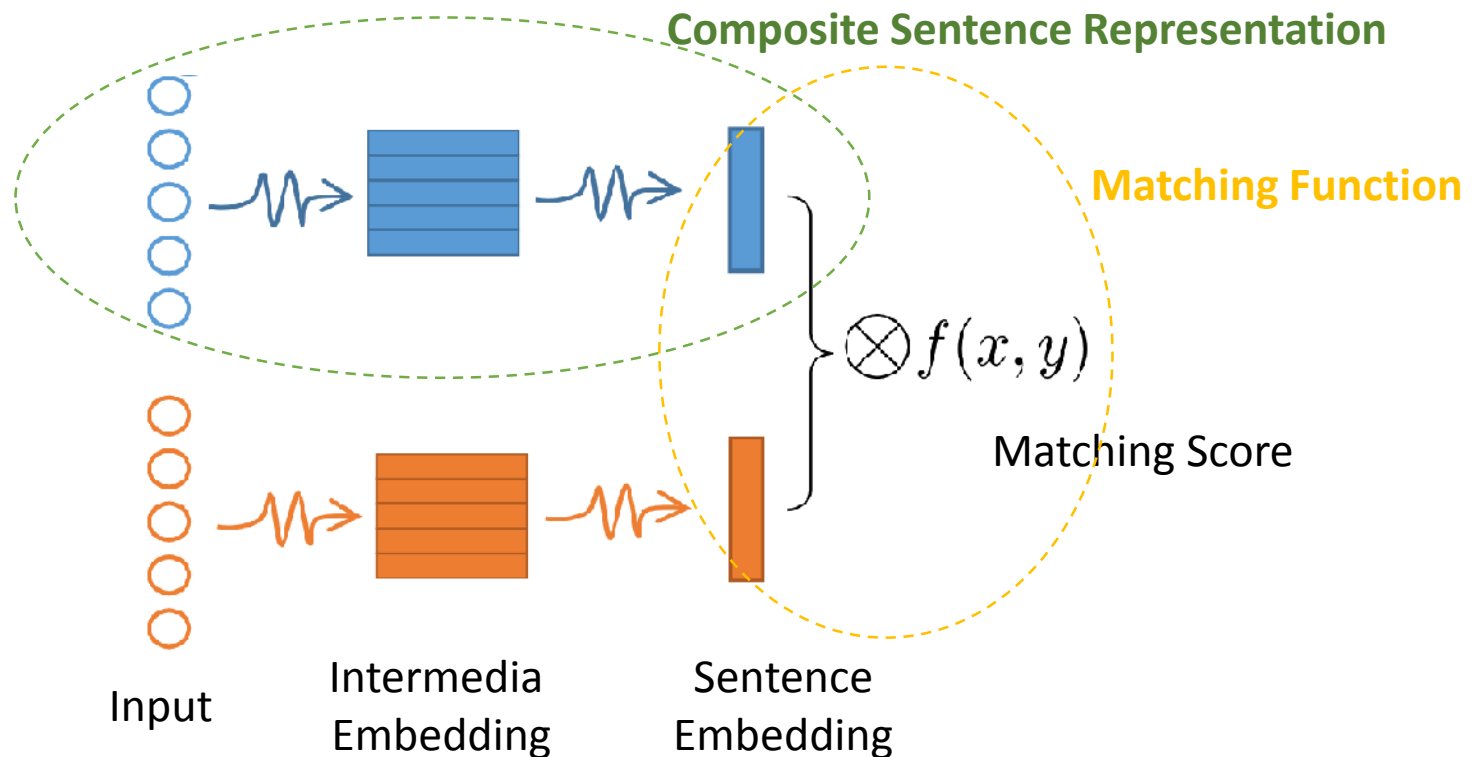
Scoring function based on
the interaction between texts

Map each text to a
representation vector

- Representation Based Models
- Interaction Based Models
- Best Single Deep Model

Representation Based Models

- Composite each sentence into one embedding
- Measure the similarity between two embeddings

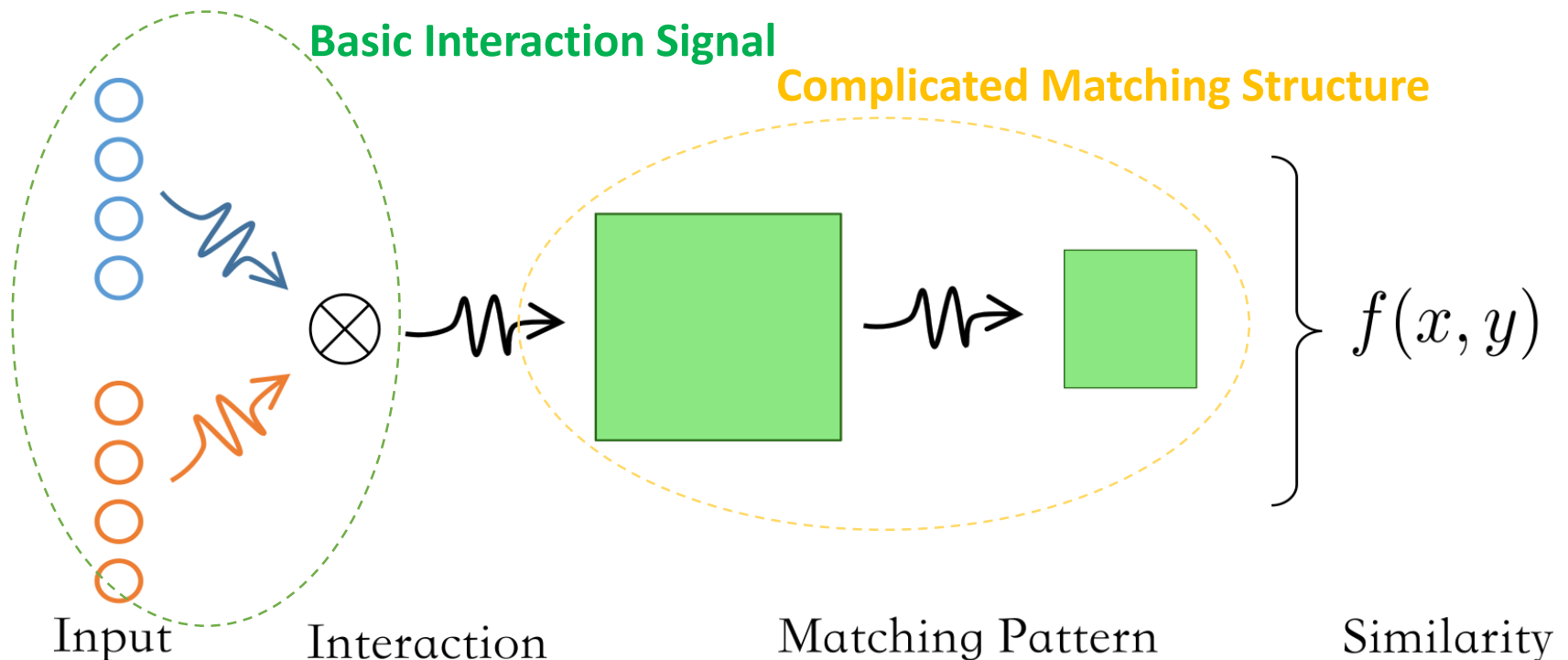


Typical Representation Based Models

- **DSSM**: Learning Deep Structured Semantic Models for Web Search using Click-through Data (Huang et al., CIKM'13)
- **CDSSM**: A latent semantic model with convolutional-pooling structure for information retrieval (Shen Y, He X, Gao J, et al. CIKM'14)
- **ARC I**: Convolutional Neural Network Architectures for Matching Natural Language Sentences (Hu et al., NIPS'14)
- **LSTM-RNN**: Deep Sentence Embedding Using the Long Short Term Memory Network: Analysis and Application to Information Retrieval (Palangi et al., TASLP'2016)

Interaction Based Models

- Two sentences meet before their own high-level representations mature
- Capture complex matching patterns

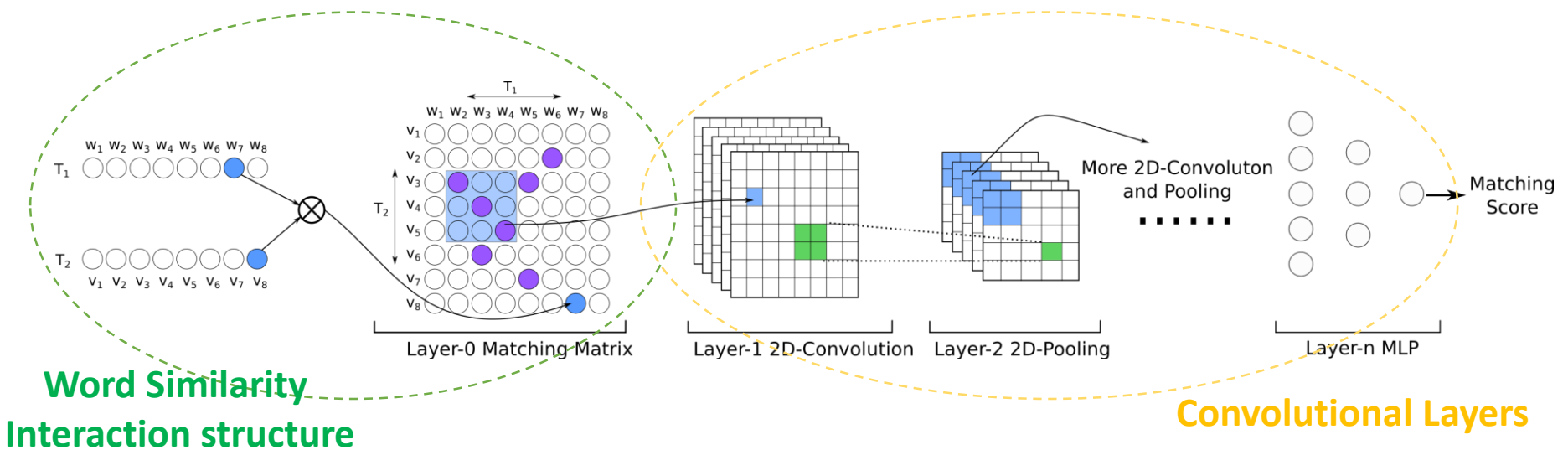


Typical Interaction Based Methods

- **DeepMatch**: A Deep Architecture for Matching Short Texts (Lu and Li, NIPS'13)
- **ARC II**: Convolutional Neural Network Architectures for Matching Natural Language Sentences (Hu et al., NIPS'14)
- **MatchPyramid**: Text Matching as Image Recognition. (Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. AAAI 2016)
- **Match-SRNN**: Modeling the Recursive Matching Structure with Spatial RNN. (Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. IJCAI 2016)

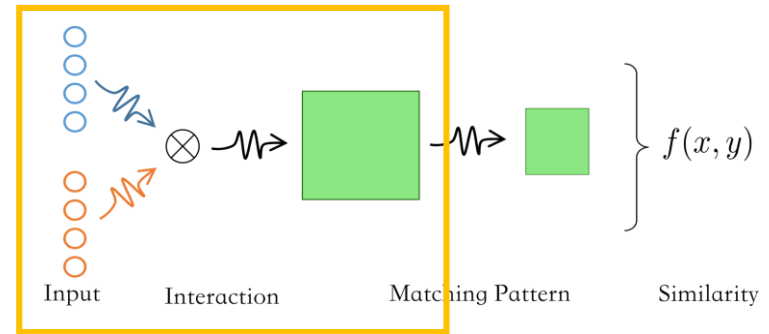
MatchPyramid

- Inspired by image recognition task
- Part 1: Construct Matching Matrix
- Part 2: Hierarchical Convolution



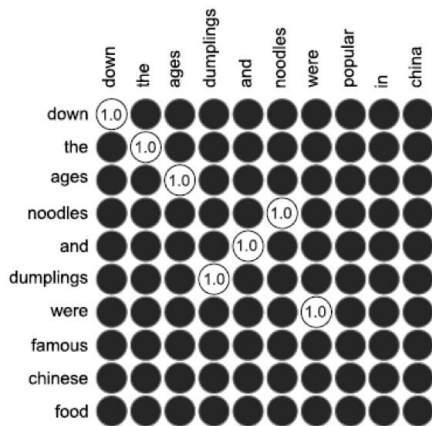
MatchPyramid

- Matching Matrix

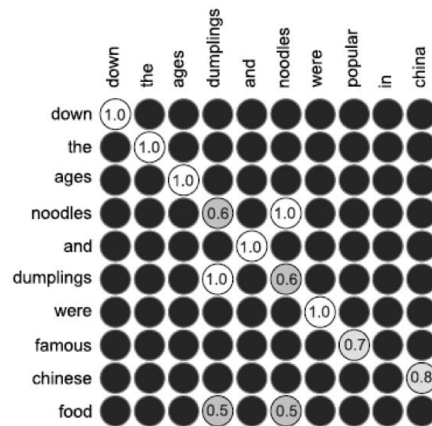


- Bridging the Gap between Text Matching and Image Recognition.

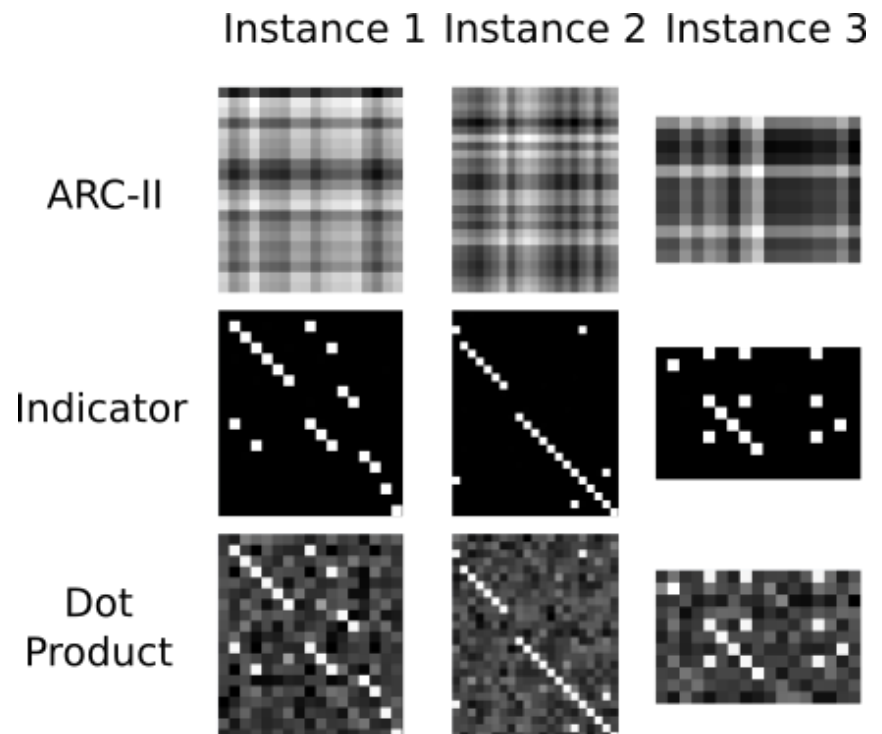
$$\mathbf{M}_{ij} = w_i \otimes v_j$$



(a) Matching Matrix-Indicator

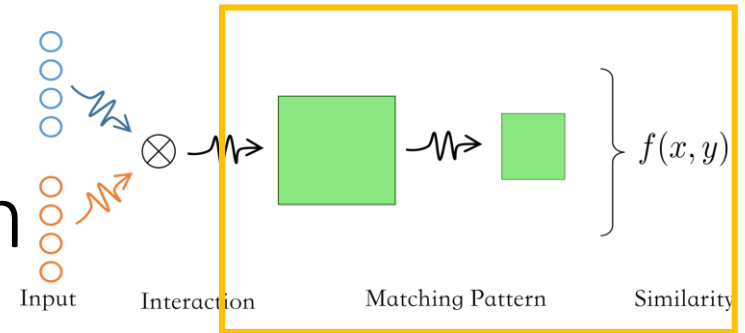


(b) Matching Matrix-Cosine

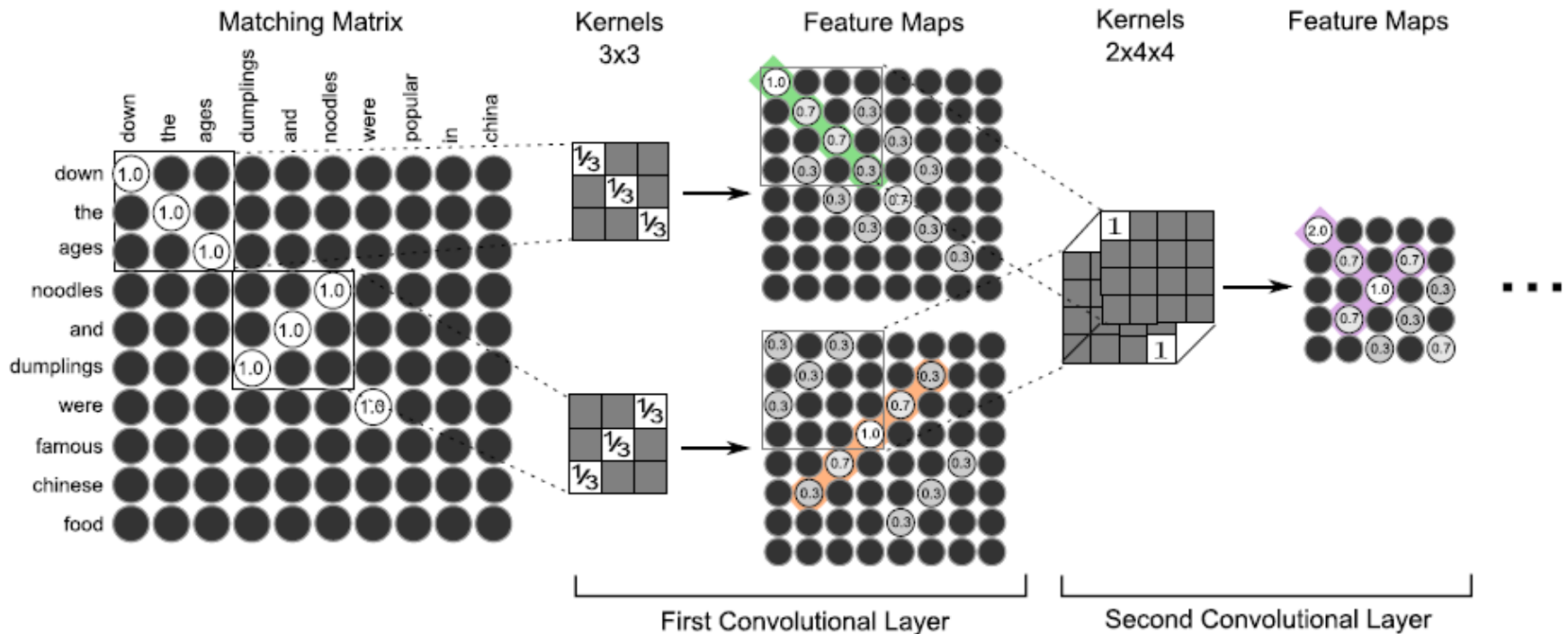


MatchPyramid

- Hierarchical Convolution

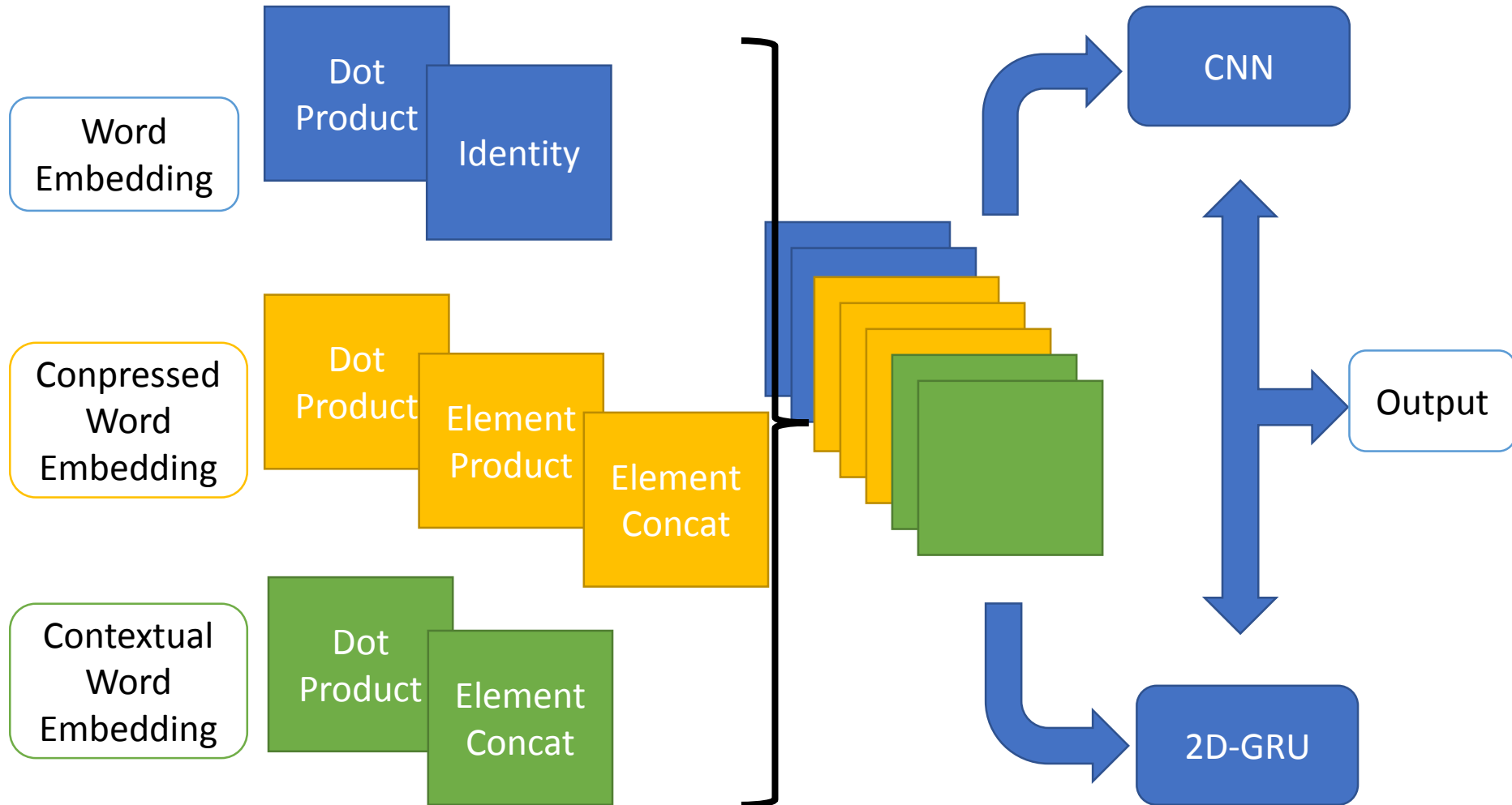


- A way to capture rich matching patterns



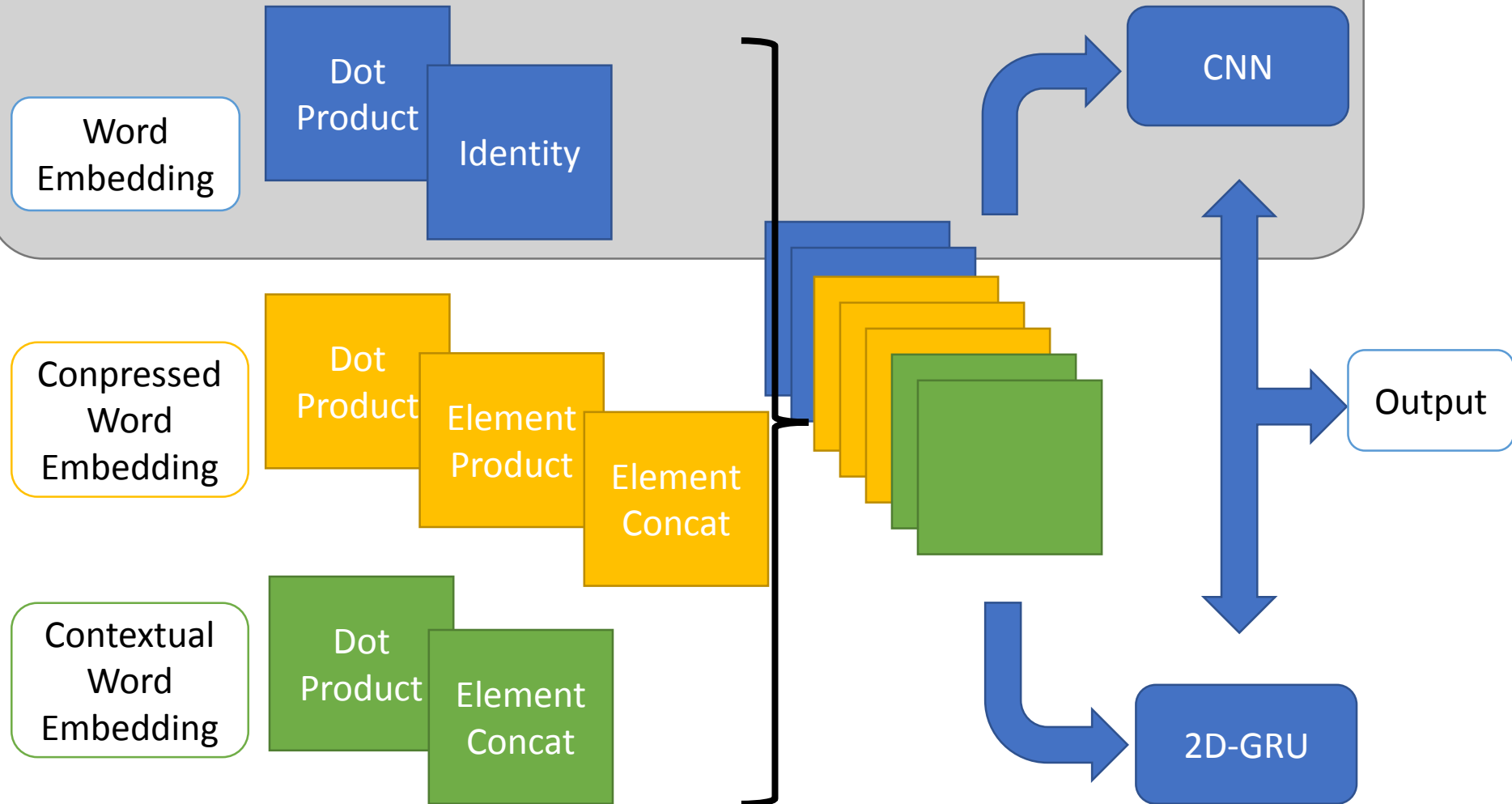
Best Single Deep Model

Combine of MatchPyramid, Match-SRNN and MVLSTM.



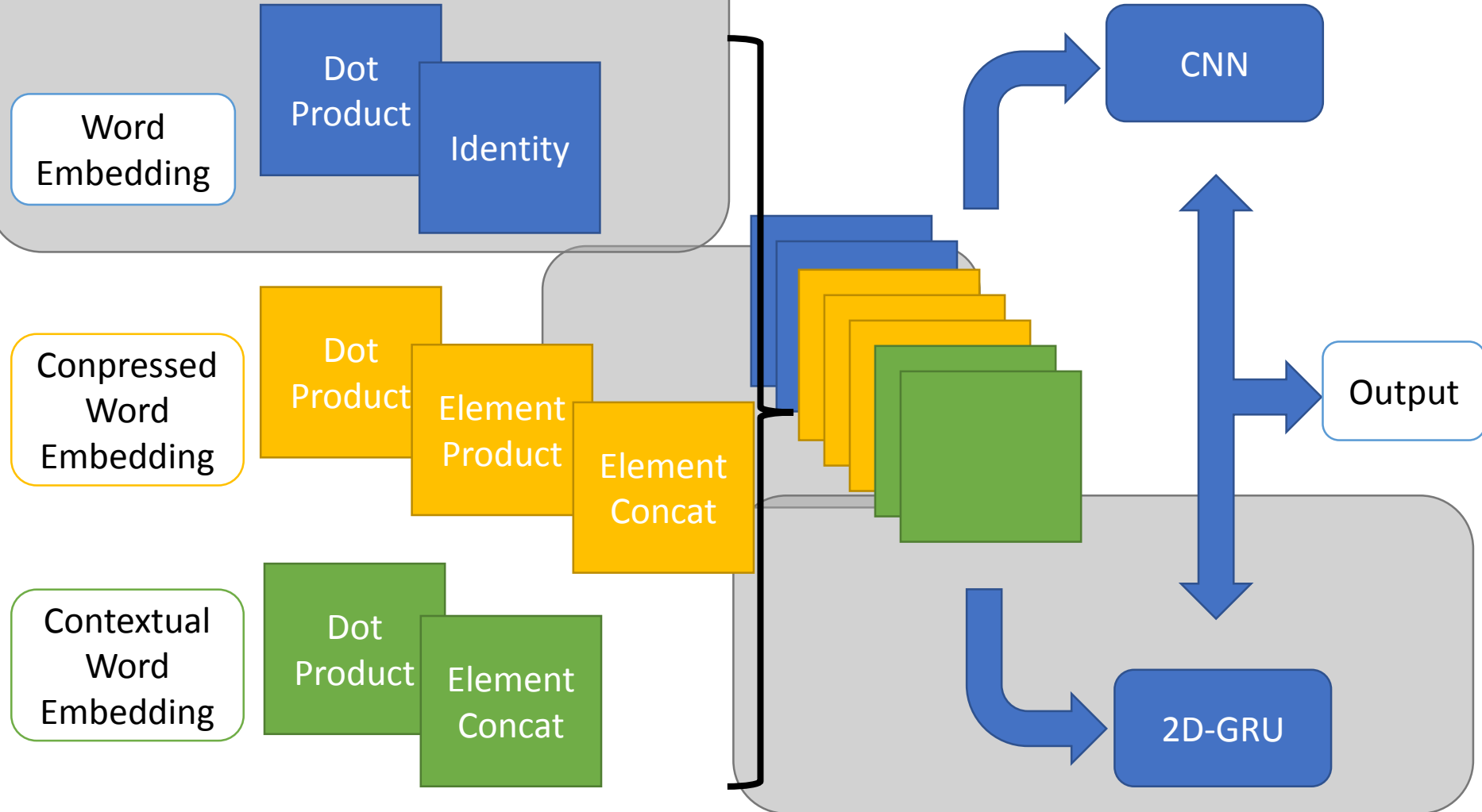
Best Single Deep Model

MatchPyramid

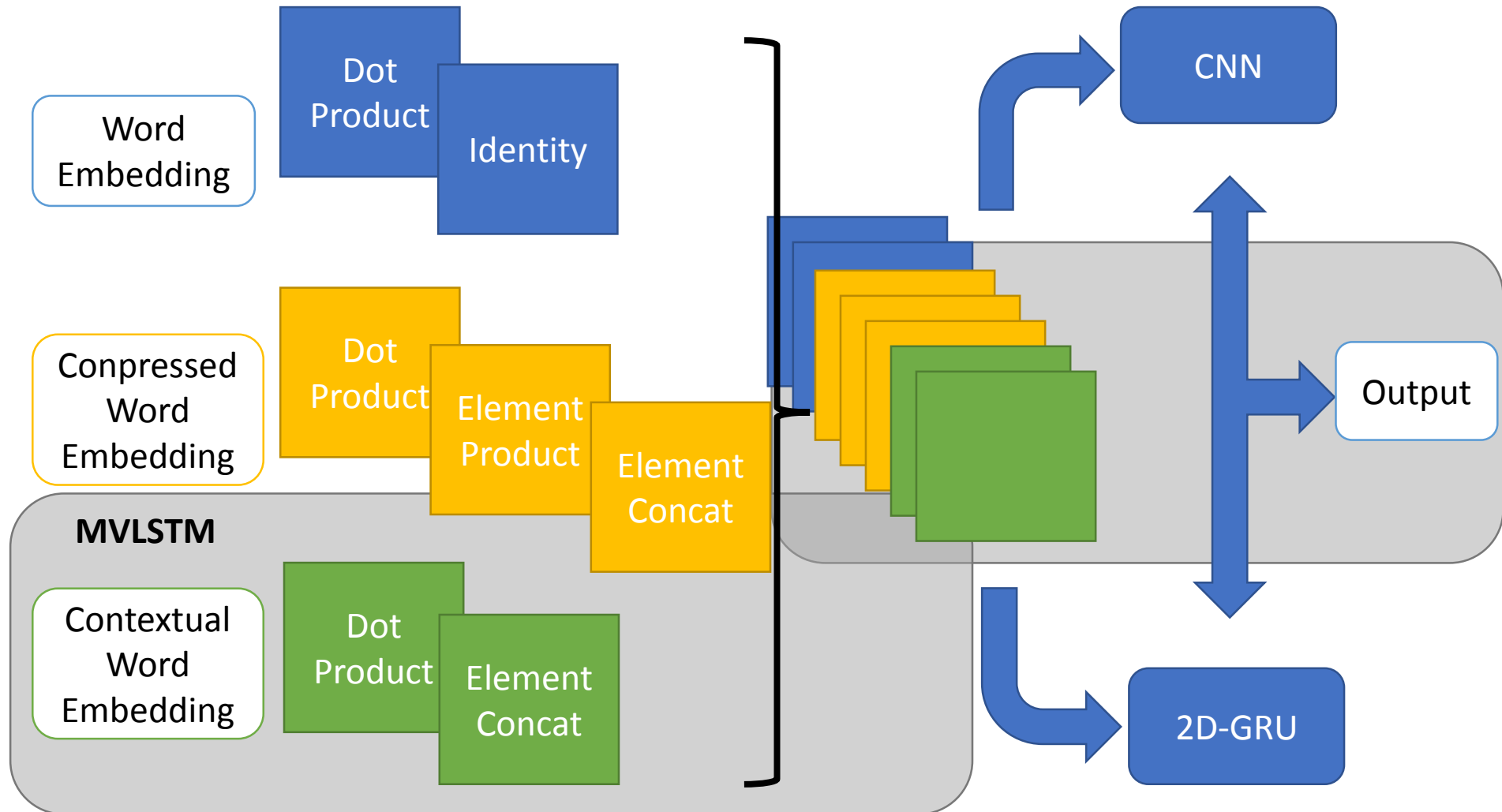


Best Single Deep Model

Match-SRNN



Best Single Deep Model

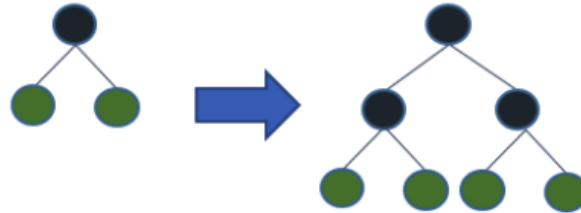


Content

- Introduction
- Solution
 - Pre-processing
 - Feature-Engineering
 - Deep Model
 - **Traditional Model**
 - Stacking
 - Post-processing
- Conclusion

Boosting Models

- XGBoost(level-wise)



- LightGBM(leaf-wise)



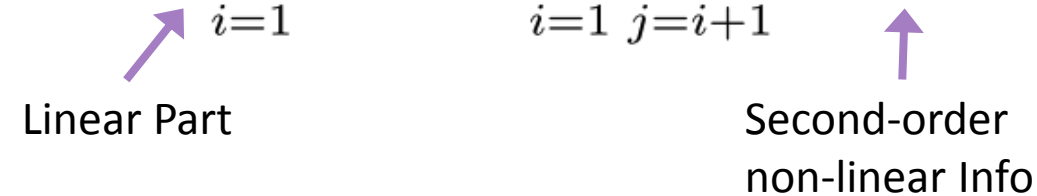
- LGB is faster than XGB without lossing performance

Boosting Models

- Feature Selection
 - Low Dimension Dense Features(1500 dim)
 - Not sensitive to high dim features and may influence training speed
- Dart Mode
 - Not sensitive to a single tree learner
 - Increase diversity of each model

Factorization Machine

$$\hat{y}(\mathbf{x}) := w_0 + \sum_{i=1}^n w_i x_i + \sum_{i=1}^n \sum_{j=i+1}^n \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j$$



Linear Part

Second-order
non-linear Info

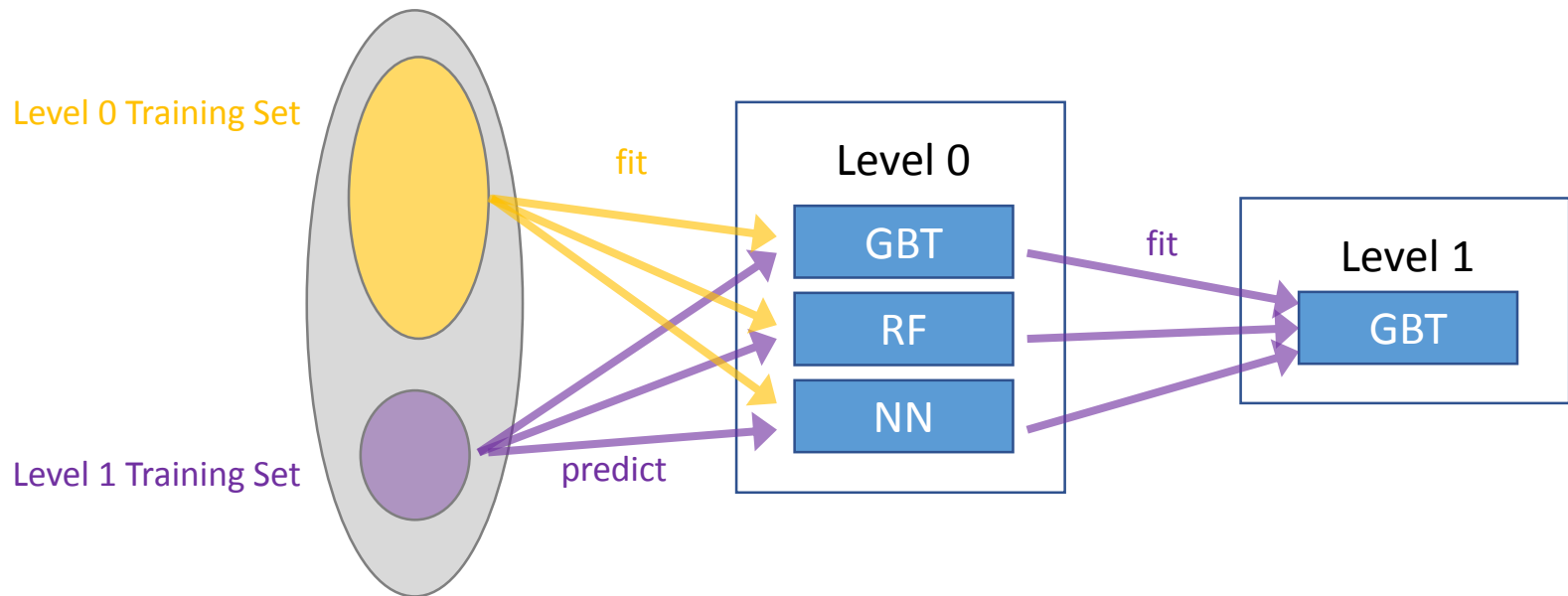
- Extract Linear and Second order interaction extra info
- Apply to 1.5 million sparse features
- Sensitive to feature scale, must normalization or hash bin.

Content

- Introduction
- Solution
 - Pre-processing
 - Feature-Engineering
 - Deep Model
 - Traditional Model
 - **Stacking**
 - Post-processing
- Conclusion

Traditional Stacking

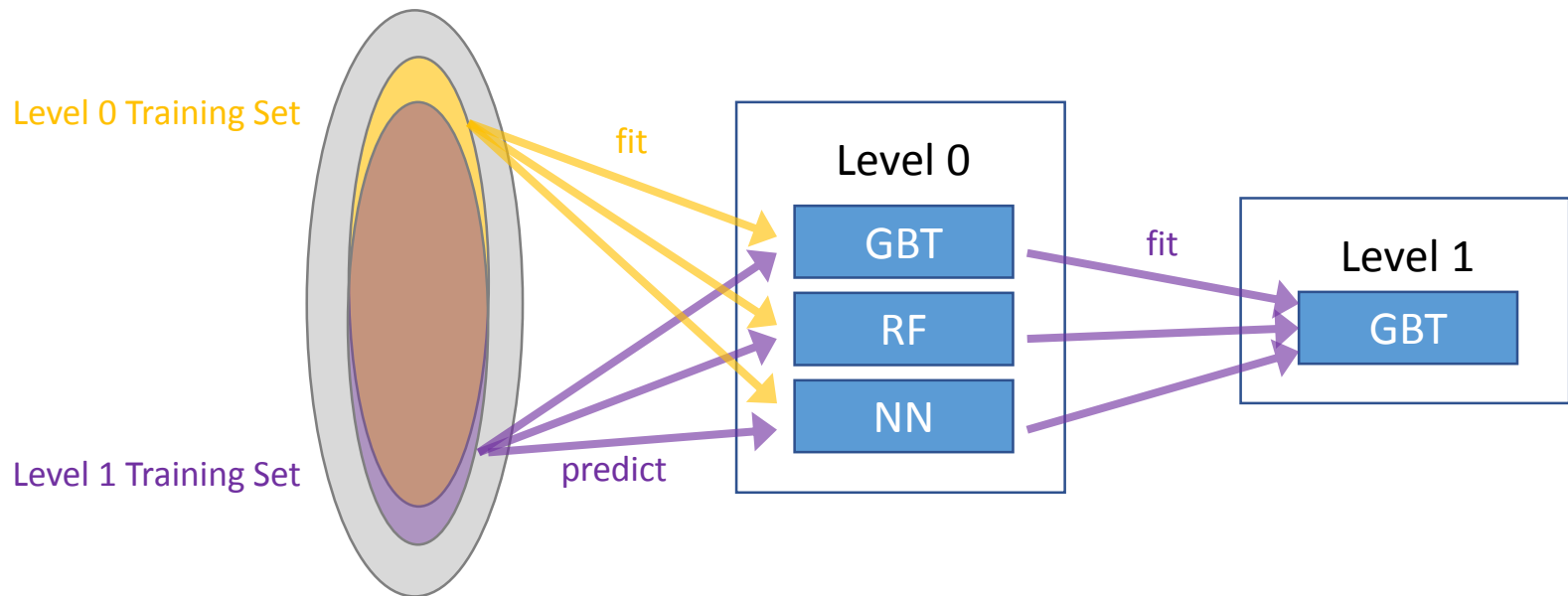
- Split the training set into two **disjoint** sets.
- **Train** several base learners on the first part and **Test** the base learners on the second part.



Deep Fusion

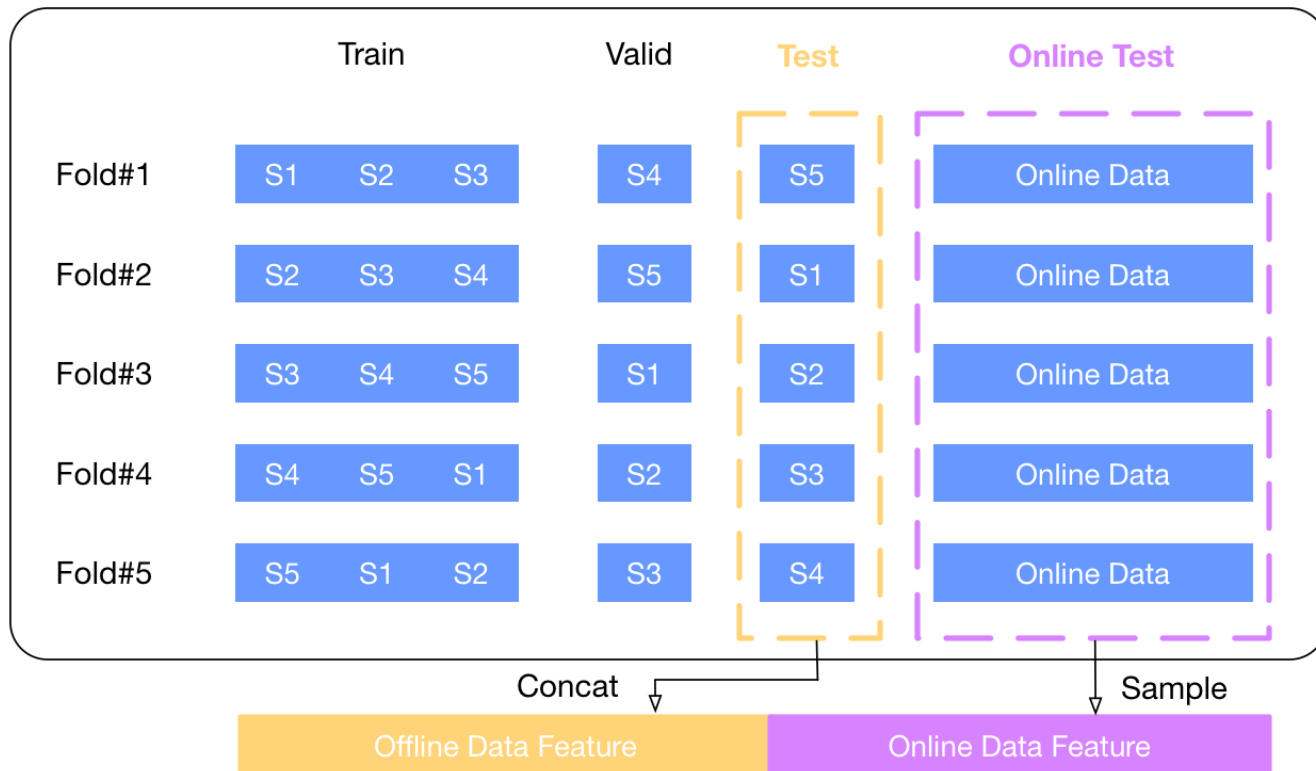
CHALLENGE

- How to **combine** good results and bad results?
- How to use information of **complete** set?



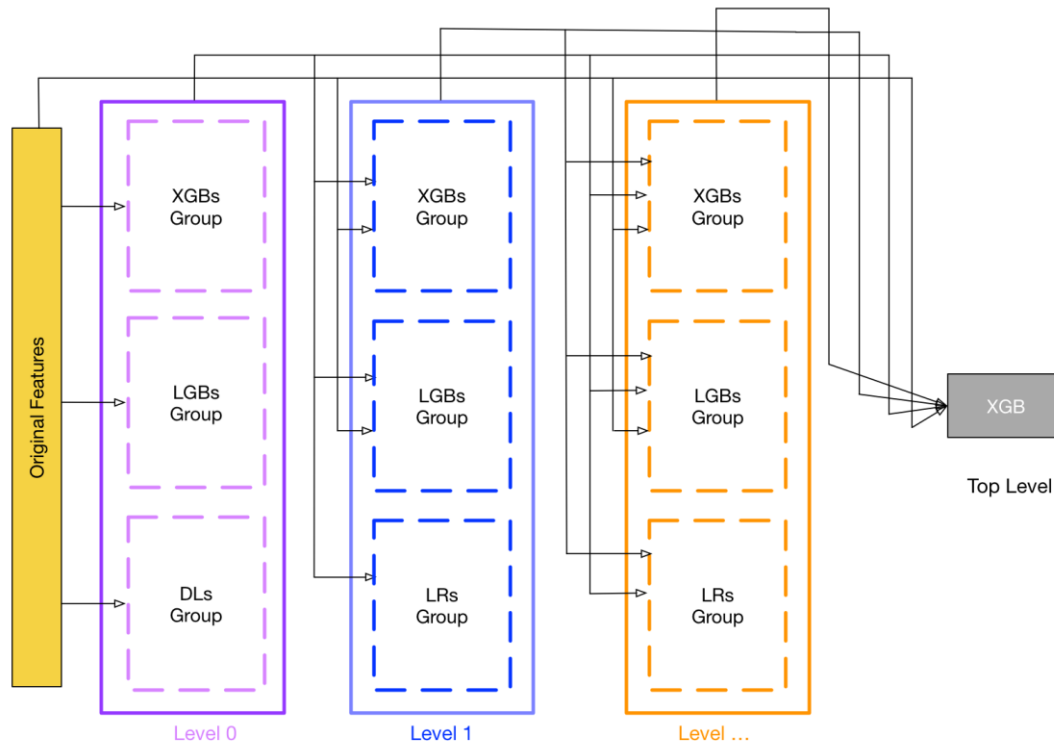
Deep Fusion • Single Stacking

1. Fit and make predictions with specified model 5 times.
2. Make prediction for online data 5 times with models generated in previous step.



Deep Fusion • Cascade Stacking

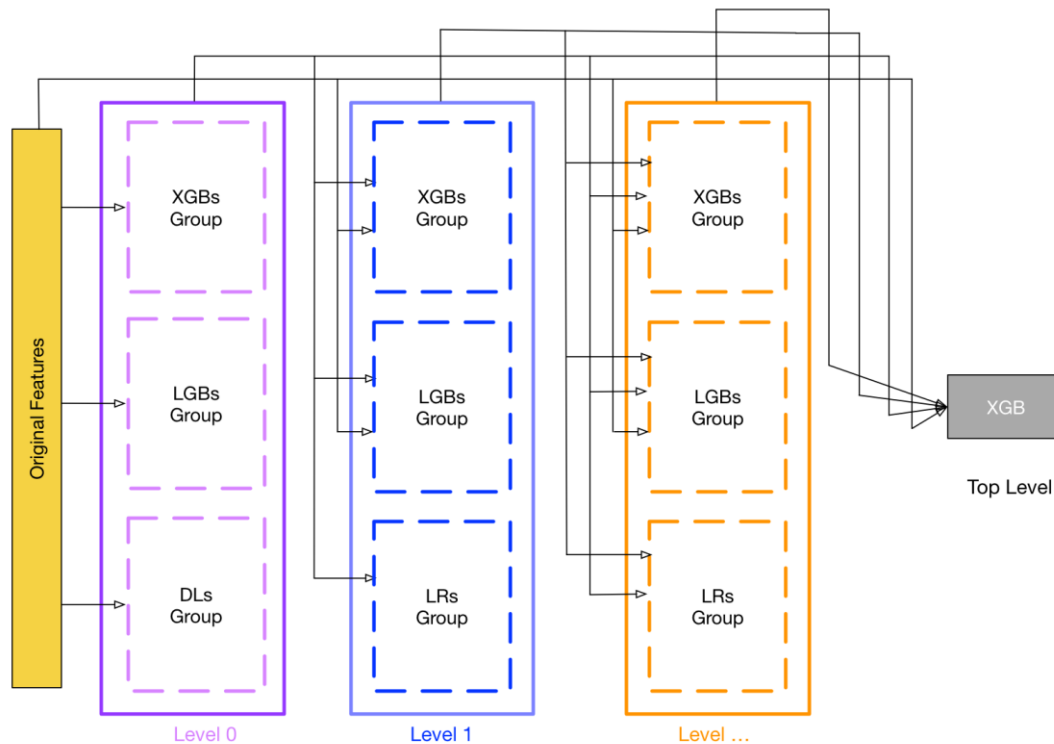
1. Fit and make predictions based on original features with diverse models.
2. Fit and make predictions based on **all** of the transformed features and **original** features with diverse models.



Deep Fusion • Cascade Stacking

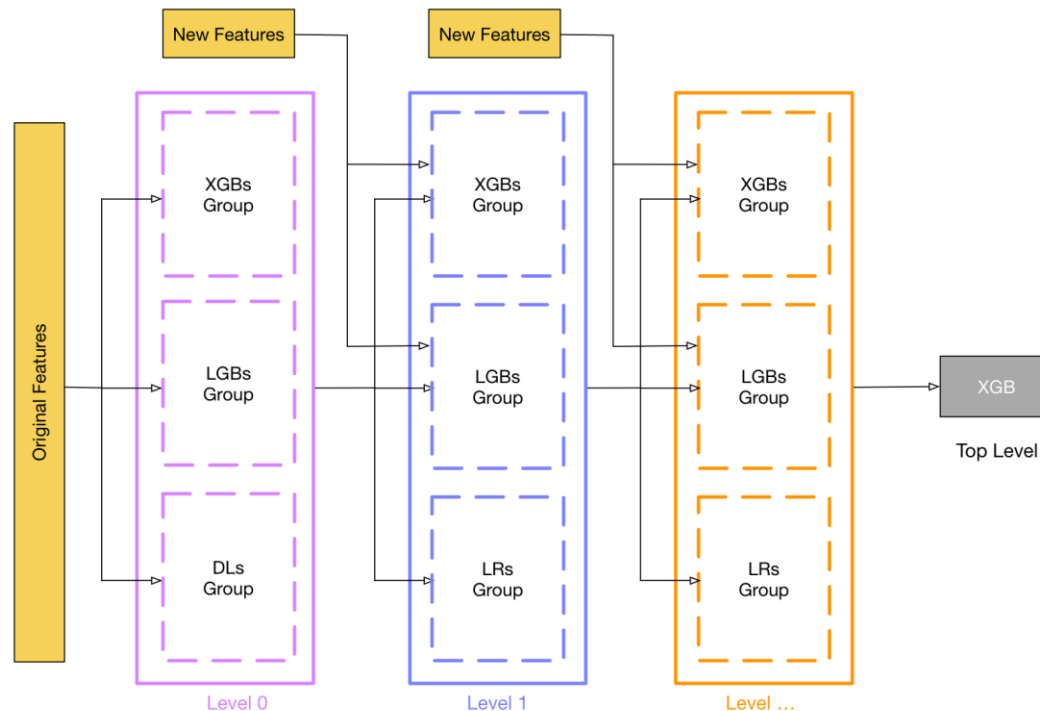
CHALLENGE

- The rebuilding process is extremely **time consuming**.
- The construction of the model is **mainly** based on the transformed features



Deep Fusion • Hierarchical Stacking

1. Fit and make predictions based on original features with diverse models.
2. Fit and make predictions based on **last layer** transformed features and **new extracted features** with diverse models.



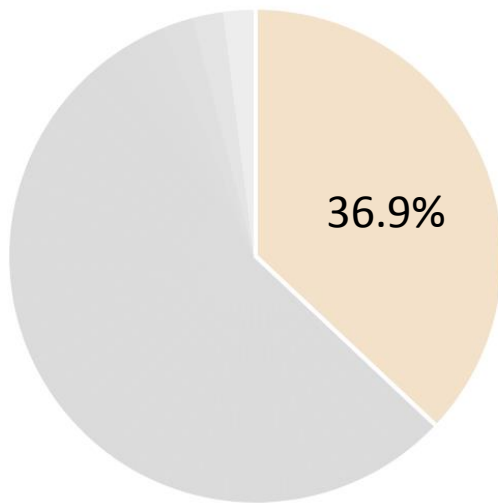
Content

- Introduction
- Solution
 - Pre-processing
 - Feature-Engineering
 - Deep Model
 - Traditional Model
 - Stacking
 - **Post-processing**
- Conclusion

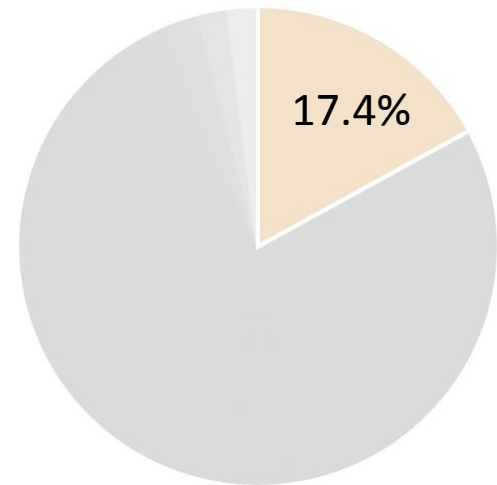
Post-processing

CHALLENGE

- The **distribution** of the offline data set (train.csv) and online data set (test.csv) are quite different.
- There are lots of **fake data points** in the online data set.



offline data set



online data set

Post-processing • Split

The standards of the division:

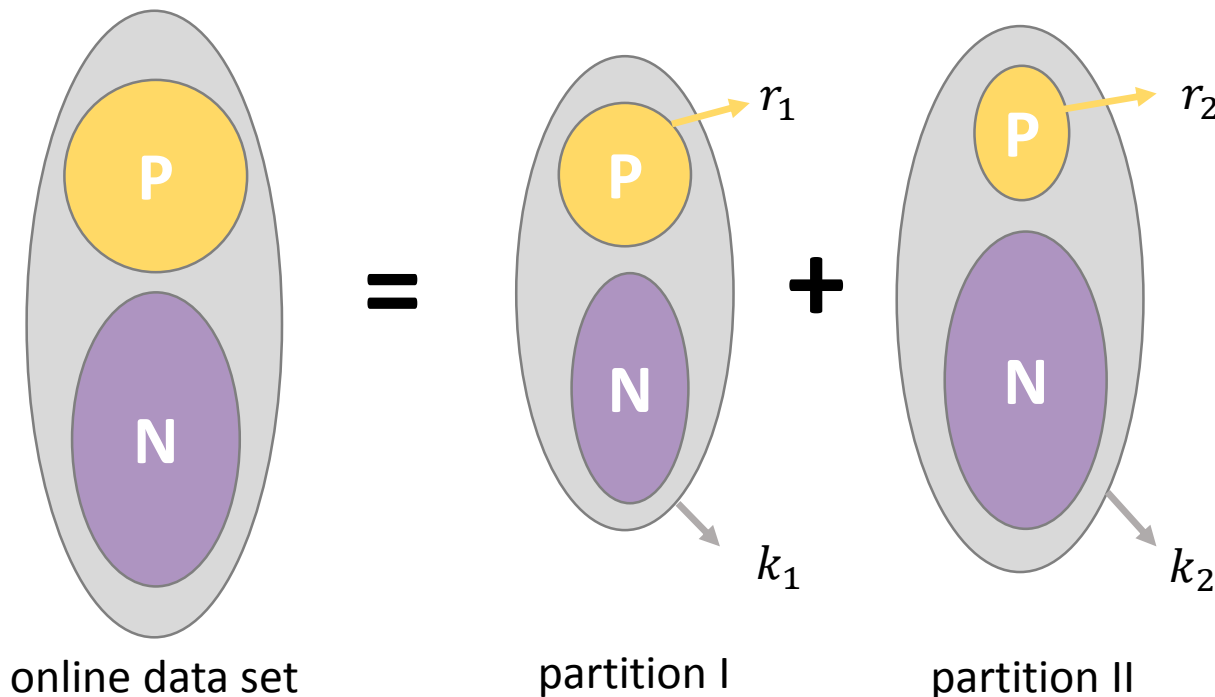
- `graph_edge_max_clique_size (mc_size)`: Number of nodes contained in the largest clique of the edge.
- `graph_edge_cc_size (cc_size)`: Number of nodes contained in the connected component of the edge.

	$mc_size < 3$ $cc_size < 3$	$mc_size < 3$ $cc_size \geq 3$	$mc_size = 3$	$mc_size > 3$
train.csv	118,065	173,164	40,482	72,579
test.csv	1,933,597	344,283	23,841	44,075

Post-processing • Split

- Solve the nonlinear equations to obtain the true data ratio and positive samples ratio of different parts for online data.

$$k_1(r_1 \log v_{11} + (1 - r_1) \log v_{10}) + k_2(r_2 \log v_{21} + (1 - r_2) \log v_{20}) = -\text{score}$$



Post-processing • Split

- Change the value of $v_{11}, v_{10}, v_{21}, v_{20}$ and corresponding scores to construct nonlinear equations.

Table 3: Data Ratio of Different Parts

	$mc_size < 3$ $cc_size < 3$	$mc_size < 3$ $cc_size \geq 3$	$mc_size = 3$	$mc_size > 3$
train.csv	29.20%	42.83%	10.01%	17.95%
test.csv	30.50%	52.19%	6.08%	11.24%

Table 4: Positive Sample Ratio of Different Parts

	$mc_size < 3$ $cc_size < 3$	$mc_size < 3$ $cc_size \geq 3$	$mc_size = 3$	$mc_size > 3$
train.csv	23.35%	14.95%	62.32%	97.26%
test.csv	5.74%	4.50%	40.88%	96.50%

Post-processing • Rescale

- Only difference between two distributions is they happen to have different proportions of positives and negatives.

$$X|(y = 0) \sim X'|(y' = 0) \quad \text{AND} \quad X|(y = 1) \sim X'|(y' = 1)$$



1st step

$$\begin{aligned} p \approx \mathbb{P}(y|x) &= \frac{\mathbb{P}(x|y)\mathbb{P}(y)}{\mathbb{P}(x)} \\ &= \frac{\mathbb{P}(x|y)\mathbb{P}(y)}{\mathbb{P}(x|y)\mathbb{P}(y) + \mathbb{P}(x|\neg y)\mathbb{P}(\neg y)} \\ &= \frac{u}{u + v}, \end{aligned}$$

Post-processing • Rescale

- Only difference between two distributions is they happen to have different proportions of positives and negatives.

$$X|(y = 0) \sim X'|(y' = 0) \quad \text{AND} \quad X|(y = 1) \sim X'|(y' = 1)$$



2st step

$$\begin{aligned} p' \approx \mathbb{P}(y'|x) &= \frac{\mathbb{P}(x|y')\mathbb{P}(y')}{\mathbb{P}(x|y')\mathbb{P}(y') + \mathbb{P}(x|\neg y')\mathbb{P}(\neg y')} \\ &= \frac{\alpha u}{\alpha u + \beta v}. \end{aligned}$$

Post-processing • Rescale

- Rescale the prediction results in different parts separately based on prior knowledge.



positive samples rate on online data set

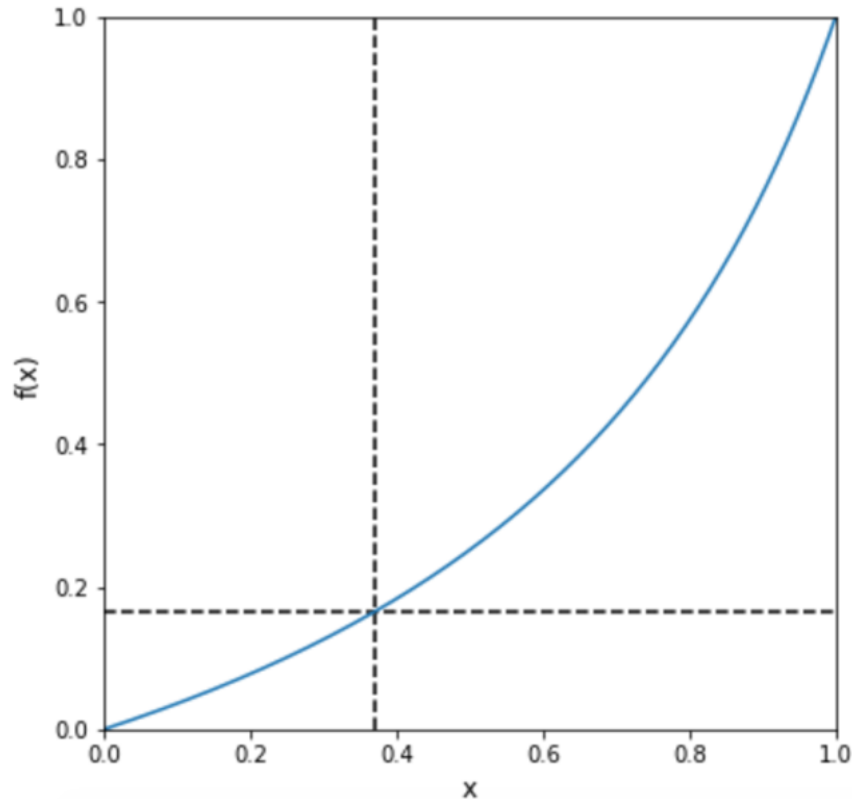
$$y_{rescale} = \frac{\frac{r_{online}}{r_{offline}} * y}{\frac{r_{online}}{r_{offline}} * y + \frac{1 - r_{online}}{1 - r_{offline}} * (1 - y)}$$

original values of prediction

positive samples rate on offline data set

Post-processing • Rescale

- Rescale the prediction results in different parts separately based on prior knowledge.



FeatWheel • Characteristics

- Developed a light weight Machine Learning framework to finish feature extraction, feature merging and so on.

S

Simple:
focus on extracting features and write a config file

F

Flexible:
specify the features you need and auto finish feature merging

E

Efficient:
generate index files to split the data set into training, validation and test

R

Reliable:
generate a separate output directory to keep the operating environment

TextNet

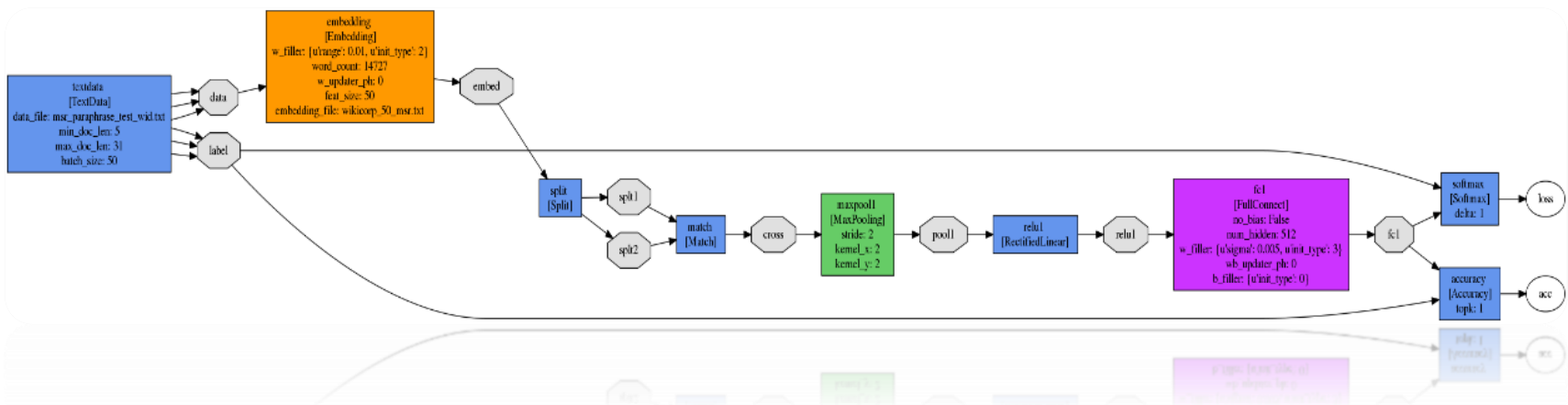
Network (DAG)

Initializer

Updater

Layer

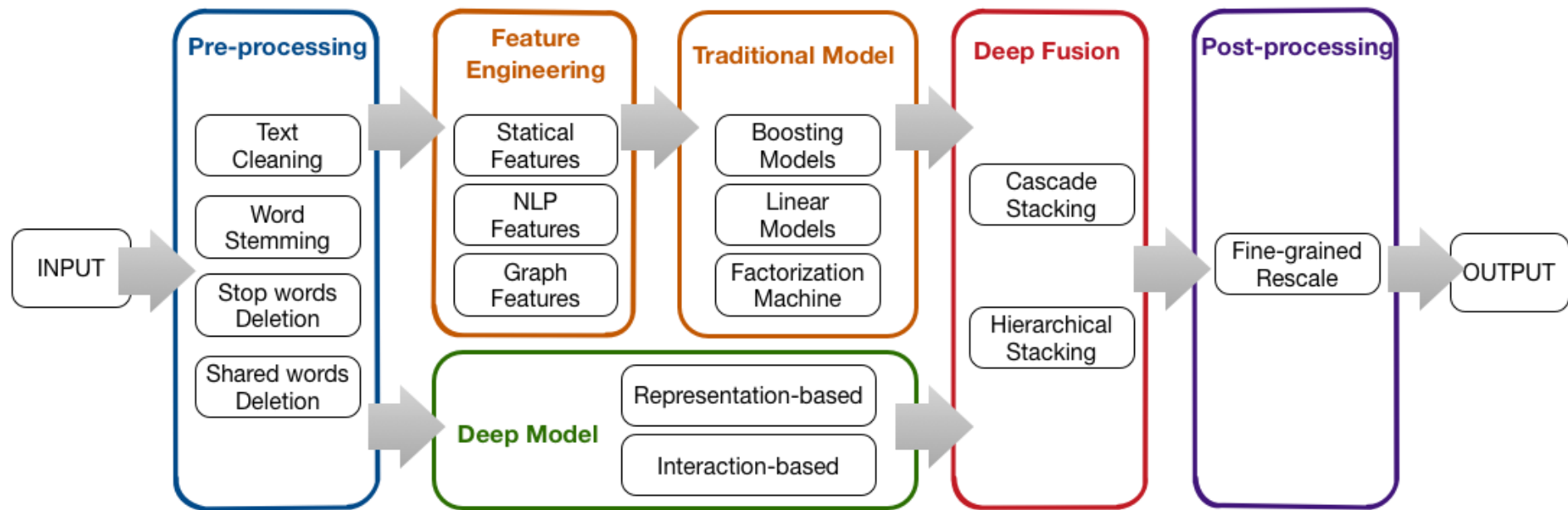
- Focus on text data, **Sparsity** and **Variance Length**.
- Support **JSON** config file to construct **DAG** networks.



Reference

- [1] <https://www.kaggle.com/c/quora-question-pairs>
- [2] <https://github.com/pl8787/textnet-release>
- [3] Pang L, Lan Y, Guo J, et al. Text Matching as Image Recognition[C]. Thirtieth AAAI Conference on Artificial Intelligence. 2016.
- [4] Wan S, Lan Y, Guo J, et al. A deep architecture for semantic matching with multiple positional sentence representations[C]. Thirtieth AAAI Conference on Artificial Intelligence. 2016.
- [5] Wan S, Lan Y, Xu J, et al. Match-SRNN: Modeling the Recursive Matching Structure with Spatial RNN[C]. Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16). 2016.
- [6] <https://github.com/HouJP/kaggle-quora-question-pairs>
- [7] <https://github.com/pl8787/textnet-release>

Conclusion



Thanks!

Q & A