Summary of Research

Zhe Xu

Shanghai Key Laboratory of Data Science

Fudan University

Email: zxu14@fudan.edu.cn

Supervisor:
Yanghua Xiao (FDU)

Outline

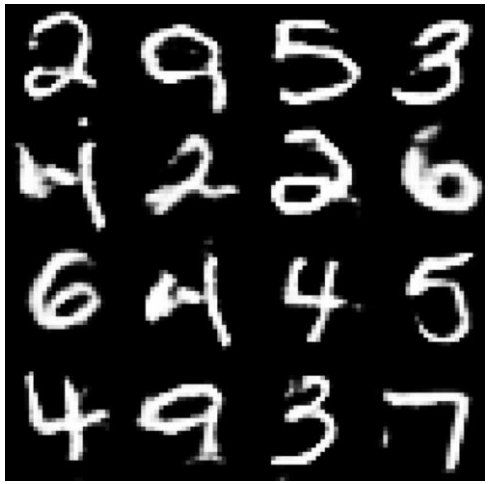
- Incorporating Complicated Rules into Deep Generative Models
- Relation Extraction: From the Perspective of Implicit Relation Reasoning
- Hierarchical Conceptual Labelling
- Entity Typing with DNN and Word Embedding

*Above all are research projects mainly. If you have any interests about my engineering projects please contact me. ☺

1. Incorporating Complicated Rules into Deep Generative Models

Problem

Results of **Generative Tasks** are far from perfect, for example:



Generated digits have fissures (3 and 6 in this image) and noise points. This image has been selected to show the problem.

"...a marble already & saies foorth priam foorth anger anger fearing norway anger mutiny noyse brought downe."

Generated Shakespeare text has word repetition.

Above all problems are **easy for human to find**. But if we want the model to learn we need to **tune the model carefully and provide much more data**.

Idea

GAN (generative adversarial network) will offer a feedback to the Generator through a network named Discriminator. Through training the Discriminator to distinguish the generated and original samples, improve the Generator at the same time.

Inspired by this, we designed another Rule Discriminator which is **trained to judge whether the results (no matter generated or original) is consistent with some Rules (e.g. No word repetition in a sentence.)**. Then we concatenated the Rule Discriminator with the Generator to improve the generated results.

It is easy for us to judge whether the results are qualified with some rules by some 'If……Else……', we called it **Rule Examiner**. See more detail in next page.

Method

- We do not have enough positive and negative samples at first and **only using Rule Discriminator to train Generator is not enough**. So the **original loss function should be maintained**.

- Since **the Rule Examiner is a part of this framework** (it need to examine the generated results in every epoch), if it works slowly (e.g. We use flood-fill algorithm to judge if there exists noise points or fissures in Hand-Written digits generation task), the training processing will be slow.

- We **do not use Rule Discriminator at first** since we do not have enough samples and **at the early training stage the generated results are useless** for Rule Discriminator to learn.(e.g. A blank image)

- The **Rules should be consistent with original loss**(e.g. Most Shakespeare text doesn't have continuous word repetition like '…anger anger…'). But we have noticed that if we let the model overemphasize the Rules like no noise points in Hand-Written digits, the generator will generate something wrapped together even like a solid white block. So we **set a hyper-parameter to adjust the loss function of Generator**.

Example1 :Hand-Written Digits

Structure

We used our framework into a Hand-Written Digits task (based on AC-GAN). And the structure of our Rule Discriminator is the same as the original Discriminator but without the Number label classification output.

Rule

In order to eliminate the noise points and fissures, we set the rule:
There should be only one white connected block. We examined it by flood-fill algorithm.

Example1 :Hand-Written Digits

Result

From the images we can find that there is less fissures after using our framework. We can get that more directly in our statistic results. **P_R is the percentage that the results can pass the Rule Examiner (We have used it as part of our framework). P_H is the percentage that human marked the result 'good'.**

Without our framework

With our framework

Metric	MNIST test data	AC-GAN	Our model
P_R	96.2%	96.5%	**99.7%**
P_H	57.1%	58.1%	**74.4%**

Statistic results

Example2: Chinese Question

Structure

Our baseline model is VAE-GAN, and the structure of Rule Discriminator is the same as the original one without a Predicate output.

Rule

Rules for Chinese question generation
1 Sentences should end with '#' (a special character).
2 The subject should appear only once in a sentence.
3 There are no continuously repeated characters in the sentence.
4 The length of sentences should be more than 4 characters. *(A Chinese question should not be too short.)*
5 The number of low frequency words should less than half of sentence length.

*We replace the subject in sentence with '~', and our training sentences all end with '#'.

Example2: Chinese Question

Result

The definition of P_R and P_H are same as the previous ones. And we show some cases with translation or annotation in the table.

With our framework, the quality of the generated text or image has been improved a lot.

Metric	Original model	Our model
P_H	51.0%	**57.0%**
P_R for all rules	63.0%	**74.2%**
P_R for rule 1	98.2%	**99.0%**
P_R for rule 2	92.8%	**96.8%**
P_R for rule 3	76.2%	**83.8%**
P_R for rule 4	**100.0%**	**100.0%**
P_R for rule 5	99.8%	**100.0%**

Predicate	Question(without R)	Question(with R)
原料	~是什么原料	你知道~需要什么原料吗
Raw material	*What is the raw material of*	*Do you know what raw material does ~ need*
编剧	谁问是的是主剧的	你知道~的编剧是谁吗
Screenwriter	*(unreadable and without ~)*	*Do you know who the screenwriter of ~ is*
别名	你知道~这的形是是么吗吗	~还叫什么
Alias	*Do you know ~***(continuous repeated words)*	*What is ~ also called*
地点	~发生在什么地方	~发生在什么地方
Location	*Where does ~ happen*	*Where does ~ happen*
作者	《知》我~的者演是谁吗	你知道~这本书的作者是谁吗
author	*(unreadable)*	*Do you know who the author of this book ~ is*

2. Relation Extraction: From the Perspective of Implicit Relation Reasoning

Problem

Relation extraction (RE) is a fundamental task for text analysis and knowledge acquisition.
We investigate how to infer the **exact relation in a knowledge base** for an entity pair from their surface patterns (relations not included in the knowledge base) in the **unstructured text**.

Idea

We believe that **sentences mentioning an entity pair certainly provide some information about the KB relation** of this entity pair, though it might not directly describe the relation.
E.g. A triple <Jobs, the-ceo-of, Apple> is in Freebase and a sentence ‘Jobs joins Apple’ is in Wikipedia. We cannot conclude the former relation from the latter sentence. But if we know more sentences like ‘Jobs created Apple’ and ‘Jobs dominated the development of Apple’, we have a higher confidence to infer that Jobs is the ceo of Apple.

Dataset Construction

1. Count the number of pairs of entities for each KB relation in KB (we use Freebase) and sort them. Chose the **α** (we select 88) commonest relations with corresponding entity pairs. Delete some meaningless entity pairs (e.g. name of the entity is a number.)

2. Extract sentences in Wikipedia containing above entity pairs. If less than **β** (we select 3) sentences mention an entity pair, this pair is discarded.

3. For each sentence, use Stanford OpenIE to extract the surface pattern between the entity pair.

4. Filter out noise surface patterns by following criterions
   - If a surface pattern(such like ‘is in’, ‘has’, ‘of’, etc.) has more than **γ** (we select 1000) entity pairs, it is meaningless.
   - Merge surface patterns into a standard pattern with past tense.(e.g. ‘go’, ‘went’, ‘is going’ → ‘went’)
   - Delete surface patterns occurring less than 3 times in Wiki since they have strange semantics.(e.g. <Jobs> *took another medical leave from* <Apple>.)
   - Delete surface patterns labelled with more than 20 KB relations since they are less discriminative.(e.g. <Jobs> *left* <Apple>.)

Model

r: set of all surface patterns in the training dataset.
R: set of all KB relations in the training dataset.
t = <*e*1, *e*2>: an entity pair in the training dataset.

r_j: the *j*th surface pattern in ***r***.
R_i: the *i*th KB relation in **R**.

The weight matrix **θ** of Dense Layer represents the contribution offered from a surface pattern to a KB relation. (e.g. $\boldsymbol{\theta}_{ij}$ quantifies how much contribution provided by r_j to infer the KB relation R_i)

If *t* has KB relation R_i, then *t* is an *instance* of R_i, and < *t*, R_i > is an ***observed fact***. If *t* has surface pattern r_j in the training dataset, then *t matches* r_j. If ***t* is an instance of R_i, and matches r_j**, then ***r_j expresses R_i***.

$\vec{X}$: the ***indicator* vector of surface patterns** for entity pair *t.* E.g. $(\vec{X})_j = 1$ if the entity pair *t* matches r_j, otherwise $(\vec{X})_j = 0$.

To find the contribution matrix **θ** that maximizes the likelihood conforming the data, that is:

$$\max_{\boldsymbol{\theta}} \prod_{<t,R_i>\in\mathcal{O}_F} P(R_i|t,\boldsymbol{r}(t),\boldsymbol{\theta})$$

where $P(R_i|t, \boldsymbol{r}(t), \boldsymbol{\theta})$ is defined as:

$$P(R_i|t,\boldsymbol{r}(t),\boldsymbol{\theta}) = \frac{\exp(\vec{\boldsymbol{\theta}}_i^T \vec{\mathbf{X}})}{\sum_{l=1}^{N} \exp(\vec{\boldsymbol{\theta}}_l^T \vec{\mathbf{X}})}$$

Priori Knowledge Constraints(1)

Observation:

1. Generally, **vague surface patterns express more KB relations** than surface patterns with specific semantics.

Surface Patterns	KB Relations	Surface Patterns	KB Relations
was in	112	*co-founded*	2
was with	100	*was professor of*	2
left	78	*was entombed in*	1

2. if r_j expresses R_i with **more entity pairs**, the contribution made by r_j to infer R_i should be **larger**.

Surface Patterns	KB relations	Entity Pairs
spoke	*user.original_language*	118
university of	*education.educational_institution*	95
premiered in	*film.individual_festivals*	78
graduated from	*school_type*	77
died of	*people.cause_of_death*	70
college of	*education.educational_institution*	54

Priori Knowledge Constraints(2)

Constraint 1:

The surface pattern r_j expresses KB relation R_i through some entity pairs, we denotes **the number of these entity pairs** as H_{ij}.
we define $\Theta_{i,j}$ as:

$$\Theta_{i,j} = \frac{H_{i,j}}{\sum_{l=1}^{N} H_{l,j}}$$

Where the denominator is the total number of entity pairs that match surface pattern r_j in the training dataset. We hope **the greater $\Theta_{i,j}$ is, the greater $\theta_{i,j}$ should be**.

we apply the following constraints on the parameters:

$$(\Theta_{i,j} - \Theta_{i',j'})(\theta_{i,j} - \theta_{i',j'}) \geq 0, \quad i \neq i' \text{ OR } j \neq j'$$

To **reduce the complexity**, we compare and constrain parameters by fixing surface patterns or KB relations. That is:

$$(\Theta_{i,j} - \Theta_{i',j'})(\theta_{i,j} - \theta_{i',j'}) \geq 0, \quad i \neq i', j = j' \text{ OR } i = i', j \neq j'$$

Priori Knowledge Constraints(3)

Constraint 2:

Entities have **type** such as {person, location, organization}.
If the types of entity pairs of a KB relation are different from those of a surface pattern, then θ_{ij} should be 0.

There may be multiple types for subject (object) of KB relations and surface patterns. E.g. the types for ***talks about***'s object can be person, organization, location and so on.

Use $S(r_j)$ ($O(r_j)$) to denote the set of appropriate types for the subject (object) of the surface pattern r_j. It is same for the KB relation R_i.

For the surface pattern r_j and KB relation R_i, the type constraints are:

$$\theta_{i,j} = \begin{cases} 0, & S(r_j) \cap S(R_i) = \varnothing \text{ OR } O(r_j) \cap O(R_i) = \varnothing \\ \text{others} \end{cases}$$

Result(1)

Model performance on KBP dataset

We compared our Constrained Model and Simple Model(without prior knowledge) with Baselines: **Mintz ++** (Mintz and Bills 2009), **Hoffmann** (Hoffmann *et al.* 2011), **Surdeanu** (Surdeanu *et al.* 2012), **Han** (Han and Sun 2016).

Model	Precision	Recall	F1 value
Mintz++	0.260	0.250	0.255
Hoffmann	0.306	0.198	0.241
Surdeanu	0.249	0.314	0.278
Han	0.459	0.231	0.307
Simple Model	0.275	0.250	0.262
Constrained Model	0.248	0.449	**0.319**

Result(2)

Human Evaluation

- We compared our Constrained Model with baselines:(1) **Han**. (2) **Simple model**. (3) **Sebastian** (Riedel *et al.* 2013)
- Set a probability threshold p_0(we set 0.3) to select the KB relations with conditional probability larger than p_0. Specifically, for each entity pair *t*, find 3 KB relations with largest probabilities from the prediction results. Then delete the KB relations with probability less than *p*0.
- We define **the *score* of *t* as the sum of the probabilities of selected KB relations**. For each model, we select 100,200,500 entity pairs with the highest score as evaluation objects.
- We assigned the truth or false of each relation according to **the majority vote of the labels by three Ph.D. students**.

Precision (%)	Top 100	Top 200	Top 500	Average
Han	0.86	0.83	0.74	0.81
Sebastian	0.80	0.76	0.63	0.73
Simple model	0.77	0.73	0.62	0.71
Constrained model	**0.89**	**0.85**	**0.79**	**0.84**

Result(3)

Typical Case

It is easy for our human to find that those Surface Patterns contribute to inferring the KB Relations at the left. It proves that our method works well.

KB Relations	Surface Patterns (Contribution Weights)
tv_program.languages	*was described in* (1.33), *with the language of* (3.21), *was screened in* (2.13)
people.place_of_birth	*wrote poetry in* (0.31), *was working in* (1.87), *taught English in* (0.25), *was born in* (2.83), *grow up in* (1.23)
people.cause_of_death	*died of* (3.12), *cause of death was* (2.13), *died from* (1.09), *was diagnosed with* (0.65), *shortly succumbed to* (0.89)
person.nationality	*returned once again in* (0.91), *was working in* (1.32), *voted for* (1.78), *eventually died in* (0.79), *next king of* (0.68)
people.profession	*was fond of* (1.36), *rely on* (0.87), *was working as* (4.31), *was one of the* (3.11), *was a* (1.20), *acted on* (1.88)
film.directed_by	*was produced by* (1.39), *the baby of* (0.52), *was elaborated by* (5.89) *directed by* (7.31), *the director's name* (4.27)
people.sibling_relationship	*loved*(1.23), *the darling of*(0.98), *successor of*(1.25), *accompanied*(3.11) *lived with*(2.45)
award.honored_for	*was famous for*(1.02), *had a talent in*(2.19), *was enjoying*(1.09), *was working as*(4.21), *was interested in*(2.39), *was hero in*(0.79)
tv.tv_actor	*was a star in*(1.36), *liked*(1.09), *was busy making*(3.10), *was famous in*(3.18), *a actor in*(4.14), *a famous actor in*(5.21)

- 3. Hierarchical Conceptual Labelling

Problem

we usually adopt the bag of words (BoW) model to represent the meaning of the text. It is usually a collection of scattered words, so it is difficult for human beings or machines to understand when its size is large. It is very important to explicitly represent the semantics of a BoW.

Idea

we focus on **conceptualizing** a BoW. The use of concepts as labels, known as ***conceptual labeling***, makes a good semantic representation of a set of words. We use **Bayesian Rose Trees** to hierarchically cluster.

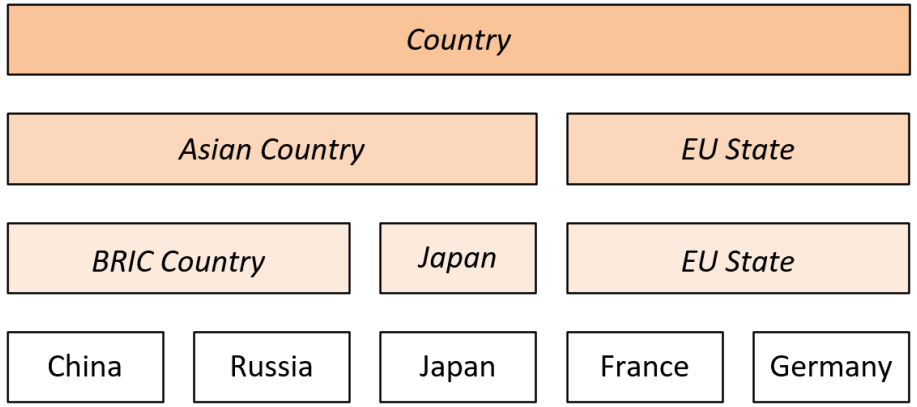
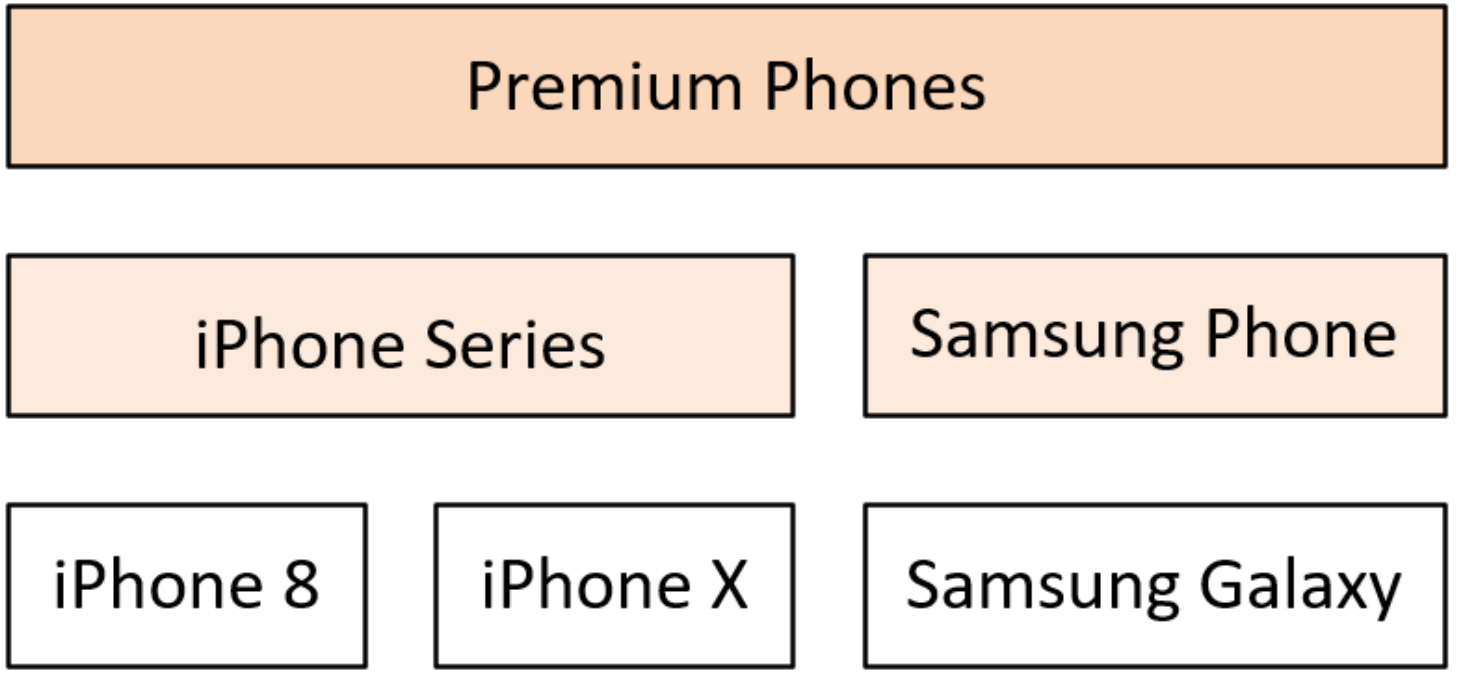


Figure: Conceptualize a Bow Hierarchically

Application

Method proposed in our paper have various applications. Especially in **interactive search**, we can use historical data to make **accurate recommendation**. E.g. If we search ‘iPhone 8’ at first, the results are iPhone 8 related. Then if we search ‘iPhone X’, our hierarchical concept could be ‘iPhone series phone’. If we input ‘Samsung Galaxy’ next time, our concept could become ‘premium phones’ **based on the Bow ‘iPhone 8, iPhone X, Samsung Galaxy’**.
Using above concept, we could improve our recommendation system greatly.



Using Probase Knowledge

Probase provides lots of ***concept entity*** pairs in the form of ***<concept, entity, frequency>***
we introduce the ***typicality*** score which is defined as (Song *et al.* 2015):

$$p(e|c) = \frac{n(c,e)}{\sum_{e_i} n(c,e_i)} \quad p(c|e) = \frac{n(c,e)}{\sum_{c_i} n(c_i,e)}$$

Define a prior probability of a concept or an entity :

$$p(c) = \frac{\sum_{e_i} n(c,e_i)}{\sum_{(c_j,e_i)} n(c_j,e_i)} \quad p(e) = \frac{\sum_{c_j} n(c_j,e)}{\sum_{(c_j,e_i)} n(c_j,e_i)}$$

Filter out noise

e.g. given the BoW {university, China, Japan, U.S.A}, it is clear that all the words are countries except ***university***, so ***University*** is likely to be noise.

Basic feature of noisy word: ***hard to be merged with any other word***.

let *D* be the input BoW. d_i and d_j be the *i*th and *j*th entity (word) in *D*, respectively.
We use $p(c|d_i, d_j)$ to measure **the degree that the concept *c* summarizes the semantics of the two entities** d_i, d_j.

$$p(c|d_i, d_j) = \frac{p(d_i, d_j|c)p(c)}{p(d_i, d_j)}$$
$$= \frac{p(d_i|c)p(d_j|c)p(c)}{p(d_i)p(d_j)}$$

Assuming that all entities in *D* have same probability, $p(d_i) = \tilde{p}$.

Introduce a hyper-parameter δ to control the ”appropriate” degree that we regard d_i as noise if it satisfies that:

$$\max_{c \in \mathcal{C}_{i,j}, d_j \in \mathcal{D}} p\left(c \,|d_i, d_j\right) < \delta$$

Bayesian Rose Trees

BRT(Bayesian Rose Trees) is a novel hierarchical clustering algorithm that **builds a tree from bottom up** by combining the **two most similar** clusters at each step.

To determine **which T_i and T_j should be merged as well as which operation should be selected**, BRT takes the criterion of maximizing the likelihood ratio $L(T_m)$, which is defined as follows:

$$L(T_m) = \frac{p(\mathcal{D}_m|T_m)}{p(\mathcal{D}_i|T_i)\,p(\mathcal{D}_j|T_j)}$$

Where $p(D_m|T_m)$ is the likelihood of data D_m given the tree T_m, which is recursively calculated as follows:

$$p(\mathcal{D}_m|T_m) = \pi_m f(\mathcal{D}_m) + (1-\pi_m) \prod_{T_k \in \mathrm{ch}(T_m)} p(D_k|T_k)$$

$f(D_m)$ is the marginal probability of data D_m. Intuitively, π_m is the prior probability that all the data in T_m is kept in one cluster instead of partitioned into sub-trees. ch(T_m) is the abbreviation of children(T_m).

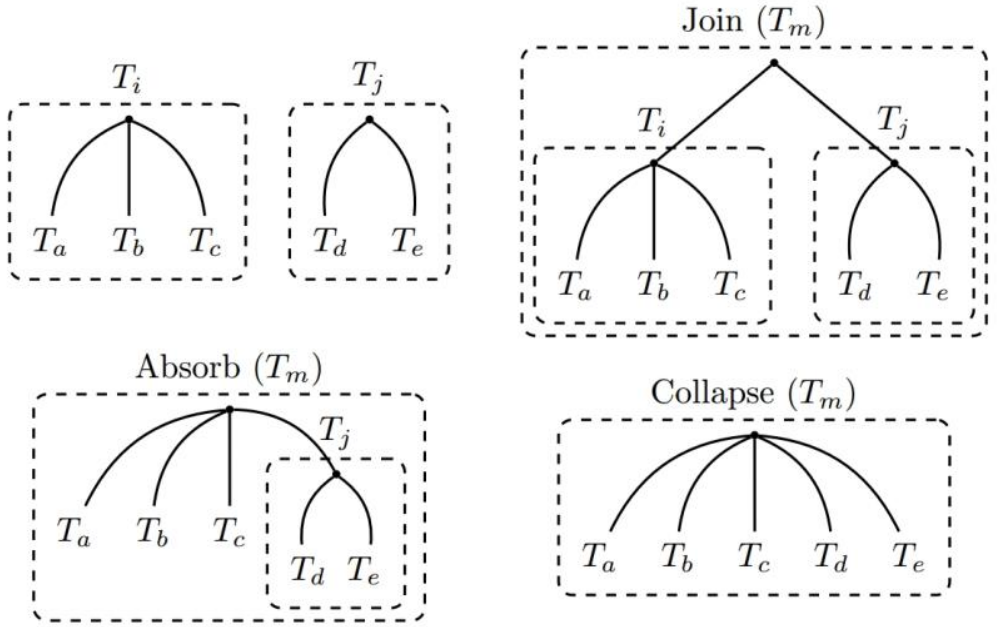


Figure: Three possible merging operations in BRT

Hierarchical Conceptual Labelling(1)

- $f(D_m)$ denotes the probability that all the data points in D_m are generated by the same probability model. D_m can be considered to be generated by concepts in Probase. Define C_m be the common concepts set of Dm. E.g. let D_m be {China,Brazil,India}, its C_m is {*developing country, emerging market, BRIC country*}.

  We define the **selected probability for c_i**, that is:

$$p_s(c_i) = \frac{p(c_i)}{\sum_{c \in C_m} p(c)}$$

  Then ***f*(*Dm*)** is computed as follows:

$$f(\mathcal{D}_m) = \sum_{c \in C_m} p_s(c)\, p(\mathcal{D}_m | c)$$

- We set the prior $\pi_m = 0.5$ since **we have no prior knowledge** to determine which partitions are more important.

Hierarchical Conceptual Labelling(2)

- The following criterion is used to select **the most appropriate conceptual label**:

$$c_m^* = \arg\max_{c \in \mathcal{C}_m} p\left(c \,|\mathcal{D}_m\right)$$
$$= \arg\max_{c \in \mathcal{C}_m} p(\mathcal{D}_m|c)p(c)$$

- We should stop clustering **when there is no appropriate label that well summarize the current cluster**. E.g. if D_m = {children, cat, cow, elephant}, we can only use labels, such as *creature*, to summarize it, which makes no sense since they are **too vague**. To avoid it, we introduce the likelihood ratio threshold γ and stop clustering when $L(T_m) < \gamma$.

Algorithm

- *D* is the clean BoW whose noise has been filtered out.
- In the initialization, we set $p(D_i|T_i) = 1$ ($i = 1, \cdots, N$) and $L(T_m) = \gamma 0$ ($\gamma 0 > \gamma$).
- In each step of clustering, the selection of the pair of trees T_i and T_j as well as the merge operation are determined by Equation 13.
- We select the concept in *Cm* that maximizes $p(c|D_m)$ as the label of the cluster D_m (see Equation 14).
- The procedure is repeated until $L(Tm) < \gamma$

Algorithm 1 Hierarchical conceptual labeling based on Bayesian rose tree

Input:

data $\mathcal{D} = \{d_1, d_2, \cdots, d_N\}$ and *LabelSet*= {}

Initialize:

number of clusters $k = N$;
$\mathcal{D}_i = \{d_i\}, T_i = \{d_i\}$ and $p(\mathcal{D}_i|T_i) = 1$ $(i = 1, \cdots, N)$;
$L(T_m) = \gamma_0$

while $k > 1$ **and** $L(T_m) > \gamma$ **do**

Find the pair of trees T_i and T_j and the merge operation that maximize the likelihood ratio:

$$L(T_m) = \frac{p(\mathcal{D}_m|T_m)}{p(\mathcal{D}_i|T_i)\,p(\mathcal{D}_j|T_j)} \tag{13}$$

Select the conceptual label c_m^*:

$$c_m^* = \arg\max_{c \in \mathcal{C}_m} p(c|\mathcal{D}_m) \tag{14}$$

Merge T_i and T_j into T_m using the selected merge operation;
$\mathcal{D}_m \leftarrow \mathcal{D}_i \cup \mathcal{D}_j$;
Add c_m to *LabelSet*;
Delete T_i and T_j, $k \leftarrow k - 1$

end while

Result on Synthetic Data—Generation

Two ways to generate Synthetic data:

1. For each BoW, selecting *m* concepts from Probase, then randomly select *n* instances from these *m* concepts' entities (ensure that **at least two entities were selected in each concept**), so we have a BoW with size *n*. Noise words are added to the BoW which are randomly selected from **other concepts' entities**.

2. For each Bow, select *m* concepts randomly in Probase and then *n* entities are selected randomly **in each concept** and *l* noise words are selected, then we get a BoW with size *mn*+*l*.

Result on Synthetic Data—Denoising

- We choose the first data generation method.
- Constitute 5 BoWs with m=2 and n=5, m=3 and n=10, m=5 and n=15(20, 25). Add 10% noise words (**noted as Set_B, are randomly selected from other concepts' entities**). Use denoising algorithm($\delta_{noise} = 5 \times 10^{-8}$) to get a **new noise word set Set_A**.
- We denote the union of difference Set_A and difference Set_B as M_{noise}, the size of the BoW(including noise words) is N_{BoW} , then the ***Accuracy*** of the denoising algorithm is defined as $(1 - M_{noise})/N_{BoW}$

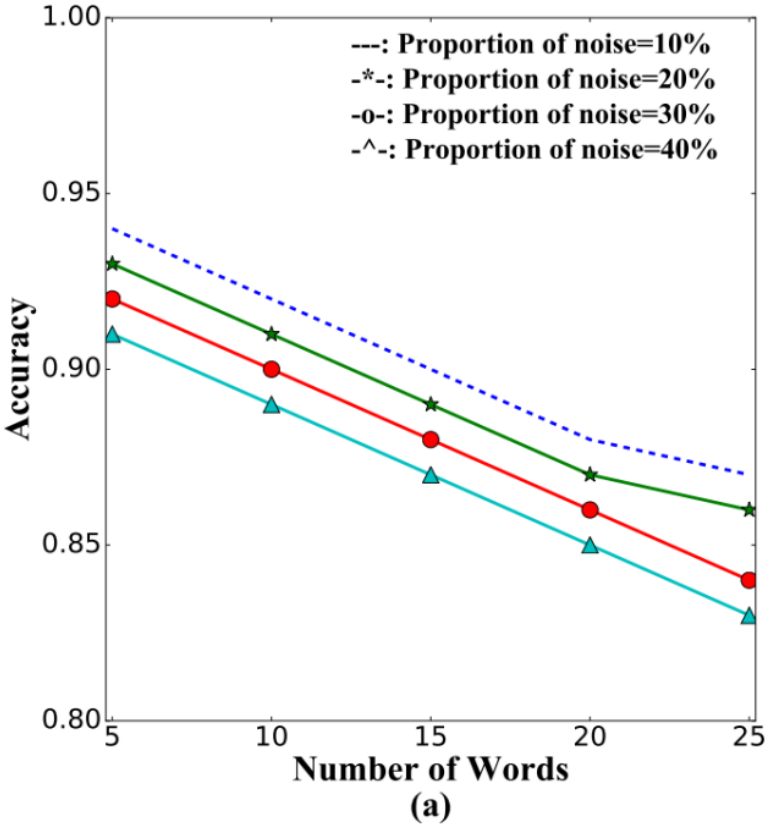


Figure: Results of denoising

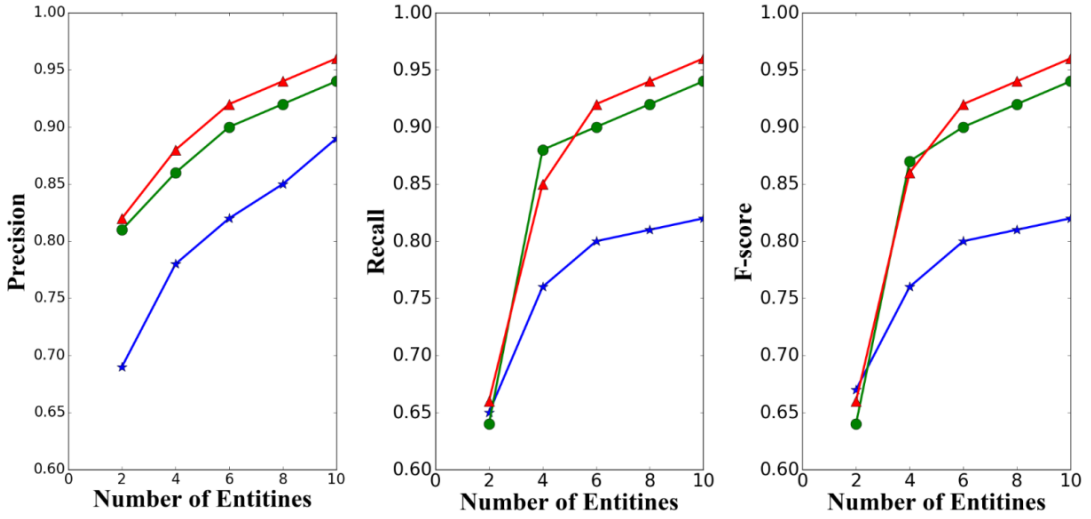
Result on Synthetic Data—Labelling

- Compare our method with the method for generating conceptual labels based on **MDL** principle and **CC**.
- Choose the **second** data generation method.
- Set L(T_m) threshold γ as 0.8. Selected **conceptual labels in the top layer** and compared them with **pre-given 5 concepts**.
- Assuming for the BoW i, the size of conceptual labels in the top layer is x_i, and y_i of them are in the pre-given label set(5 concepts).

Define: Precision: $\left(\sum_{i=1}^{t} y_i\right) / \left(\sum_{i=1}^{t} x_i\right)$

Recall: $\left(\sum_{i=1}^{t} y_i\right) / 5t$

F: $\frac{2precision \times recall}{precision + recall}$



Curve with triangle: BRT
Curve with circle: MDL
Curve with star: CC

Result on Synthetic Data—Human Evaluation

- N is the number of labels.
- We choose the **first** data generation method.
- Evaluation Standard:

Score	Description	Example
1/*N*	Appropriate	football,sport,snooker→sport
1/2*N*	General	art,painting,dance,disco→activity
0	Inappropriate	family,sibling,people→others

- Evaluate the performance (accuracy) of the algorithm by calculating the scores of the all conceptual labels.

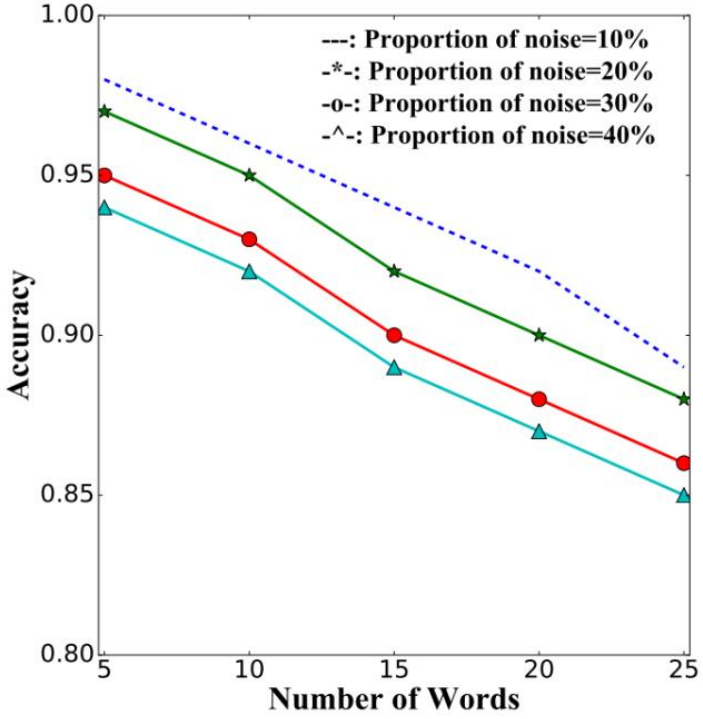


Figure: Results with different noise ratios and sizes of BOW

Result on Real Data—Human Evaluation

- Use Wiki data and Flickr data. Use LDA algorithm and extracted top words of each topic as the BoW.
- Use **manual scoring method** to evaluate the performance of the conceptual label.
- Evaluation scores on Flickr and Wikipedia data:

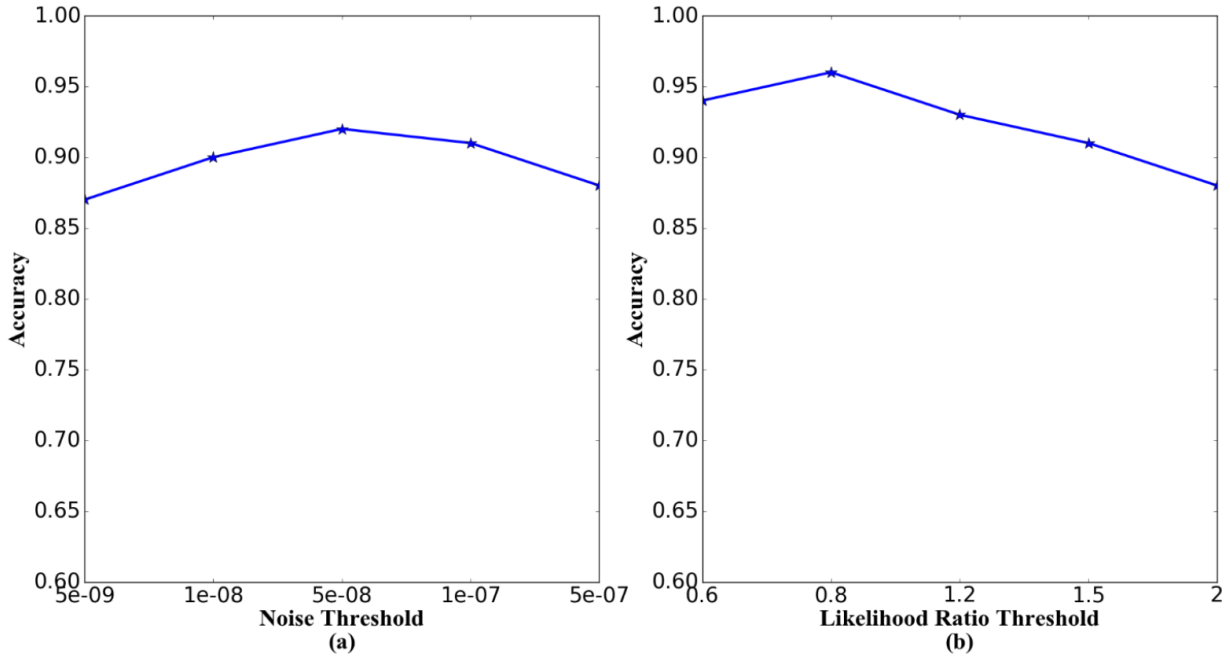
Algorithm	Flickr data	Wikipedia data
CC	0.76	0.81
MDL Method	0.85	0.88
BRT Method	0.91	0.94

Two empirical threshold

There are two empirical threshold in our Algorithm:

1. **Noise threshold** δ_{noise}.
2. **Likelihood ratio threshold** $L(T_m)$ γ

We set the size of the BoW is 10, the noise ratio is 20%, the effect of different δ_{noise} is analyzed in the figure(a):
Based on the above analysis, we set $\delta noise = 5 \times 10^{-8}$ and analyse the likelihood ratio threshold $L(T_m)$ γ in figure(b):

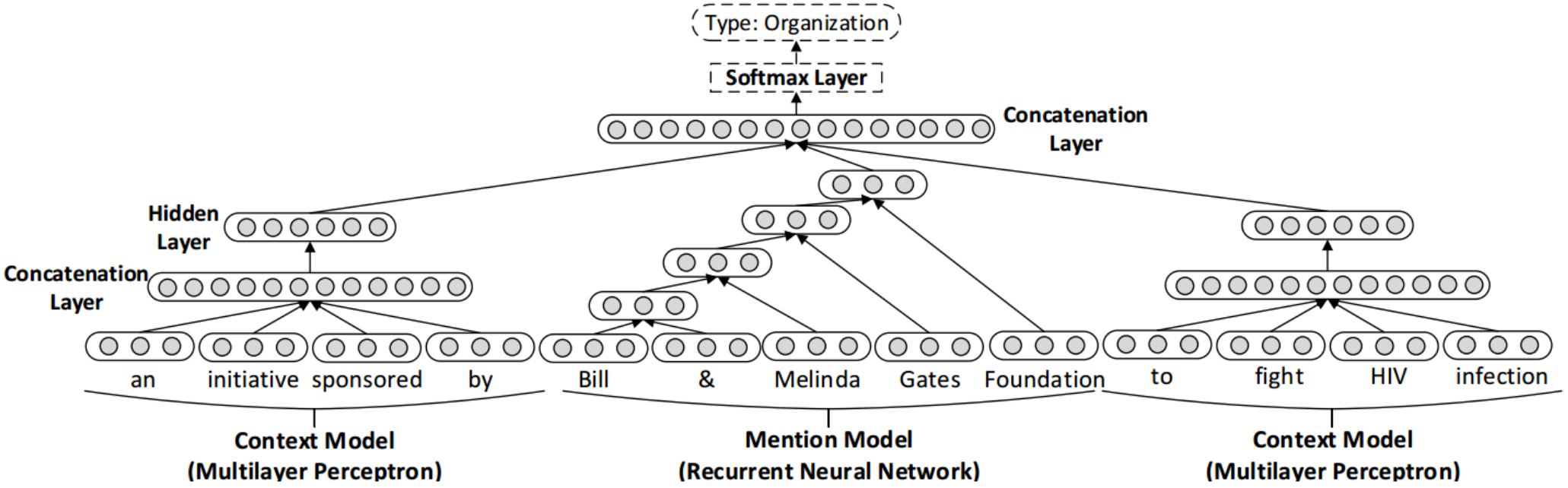


- # 4. Entity Typing with DNN and Word Embedding

*Follow Li Dong, Fu-Ru Wei, etc. A Hybrid Neural Model for Type Classification of Entity Mentions. In IJCAI, 2015.

Baseline Model

*Li Dong, Fu-Ru Wei, etc. A Hybrid Neural Model for Type Classification of Entity Mentions. In IJCAI, 2015.



It is impressive for this model to consider Context Model. However we believe that using Multilayer Perceptron and Word Embedding is not enough.

Character-level Word Embedding

*Improved from Yanyao Shen, Hyokun Yun, etc. Deep Active Learning for Named Entity Recognition. On arxiv.

- We use Bidirectional LSTM to learn more knowledge than CNN in Shen's paper.
- Character-level Word Embedding can be a good supplement to typical Word Embedding.

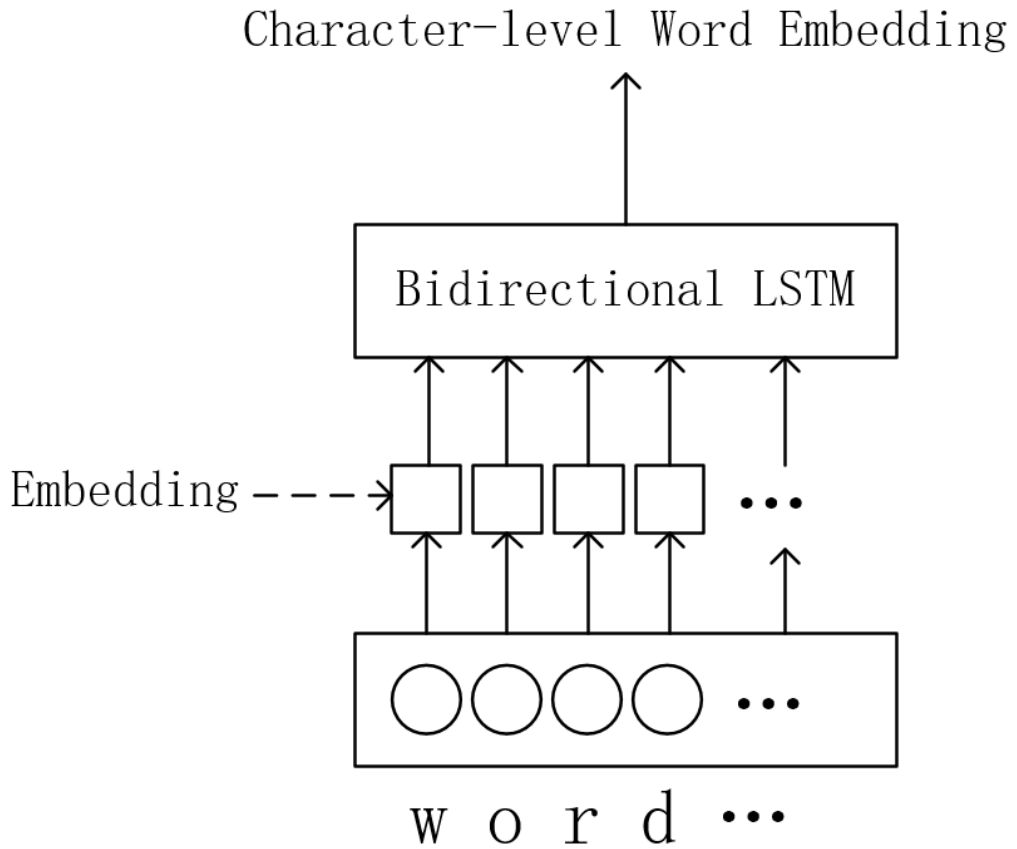
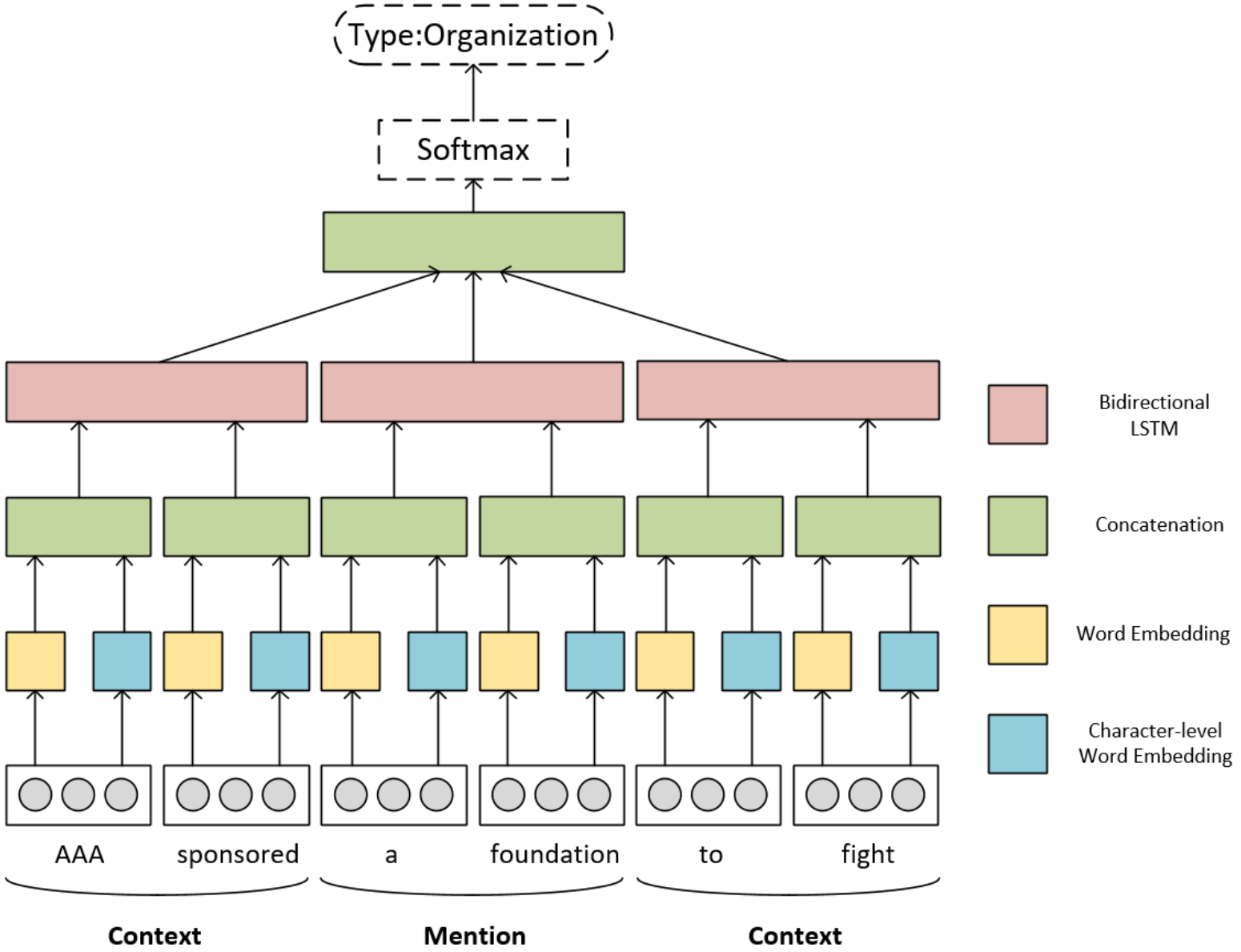


Figure: Character-level Word Embedding structure

Model

- Replace Multilayer Conception with Bidirectional LSTM.
- Add improved Character-level Word Embedding.
- The length of Context and Mention Windows can be adjusted. We choose Mention Window = 6 and Context Window = 7 to balance accuracy and training speed.



Result

- Do experiment on dataset Wiki, BBN, and OntoNotes.

- The results are as follows.

	Wiki		OntoNotes		BBN	
	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1
Baseline	0.409	0.417	0.288	0.272	0.591	0.606
Our model	**0.602**	**0.663**	**0.454**	**0.421**	**0.632**	**0.658**

Thanks !

If you have any questions or interests, please contact me without hesitation. ☺

Zhe Xu

Shanghai Key Laboratory of Data Science

Fudan University

Email: zxu14@fudan.edu.cn