Data Science for Document Analysis and Understanding

Team Member: XIAO Rudan

GE Chongjian

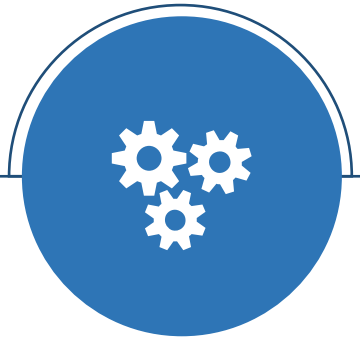
LIANG Mingzhu

2020-1-10

Presentation Contents

01 Background

- NLP-related tasks
- Vector space model
- LSTM

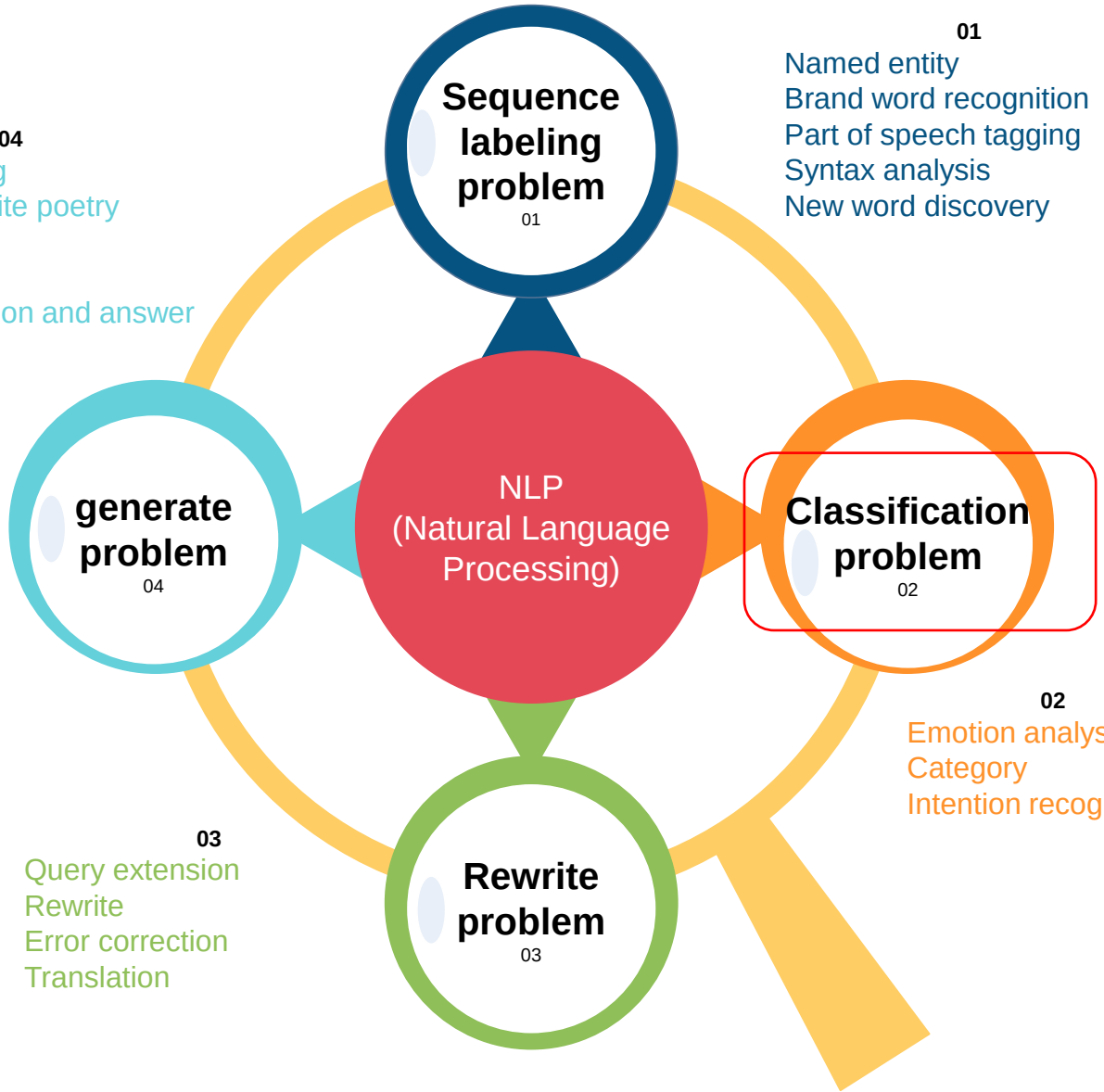


02 Methods & Result

- XML annotation
- Word frequency analysis
- Emotion analysis

https://github.com/medxiaorudan/NLP_AMMI_Emotional_Scoring

Background — NLP related task



01
Named entity
Brand word recognition
Part of speech tagging
Syntax analysis
New word discovery

02
Emotion analysis
Category
Intention recognition

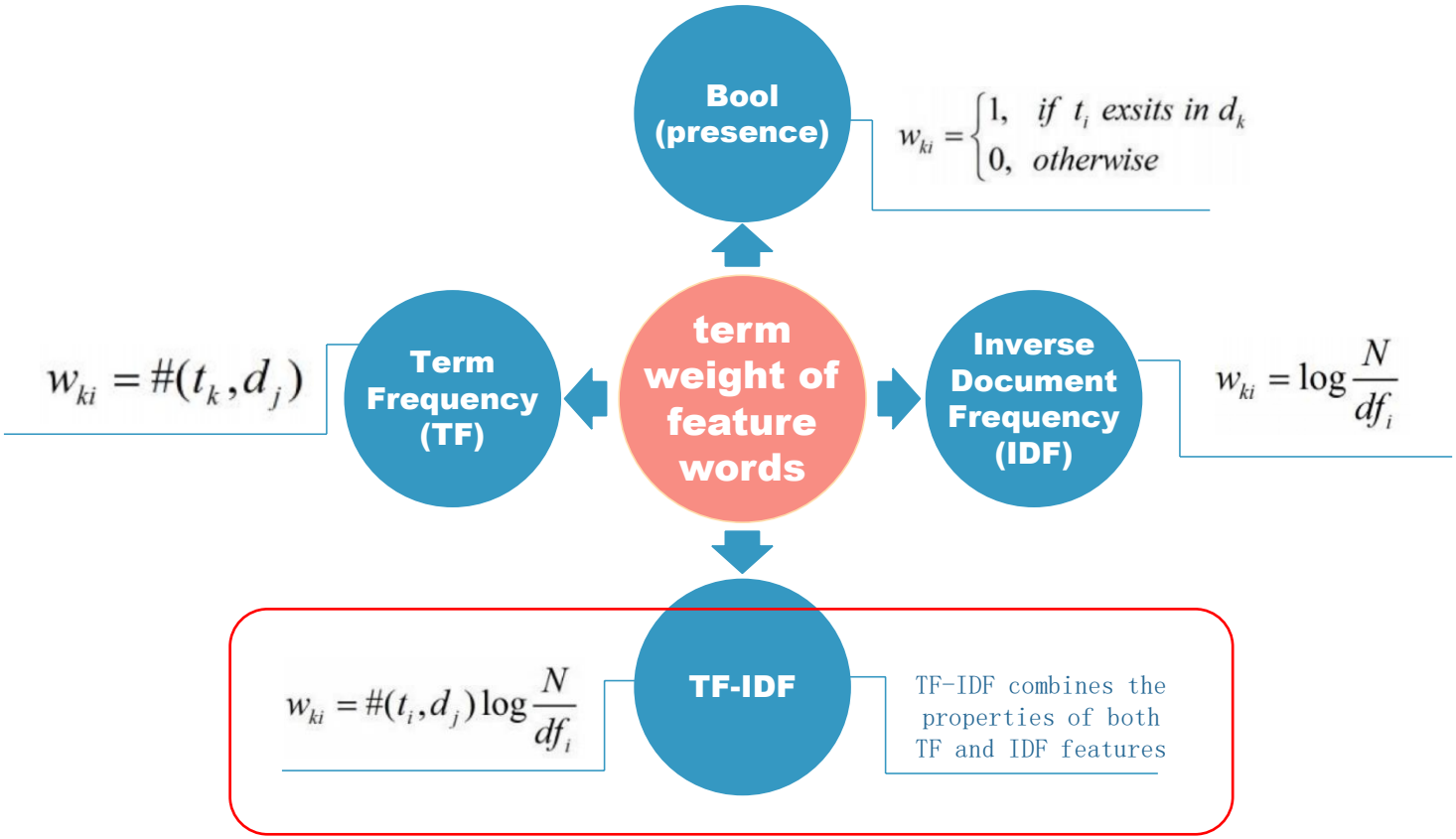
03
Query extension
Rewrite
Error correction
Translation

04
Automatic writing
Automatically write poetry
Text summary
Chat robot
Automatic question and answer
Voice synthesis

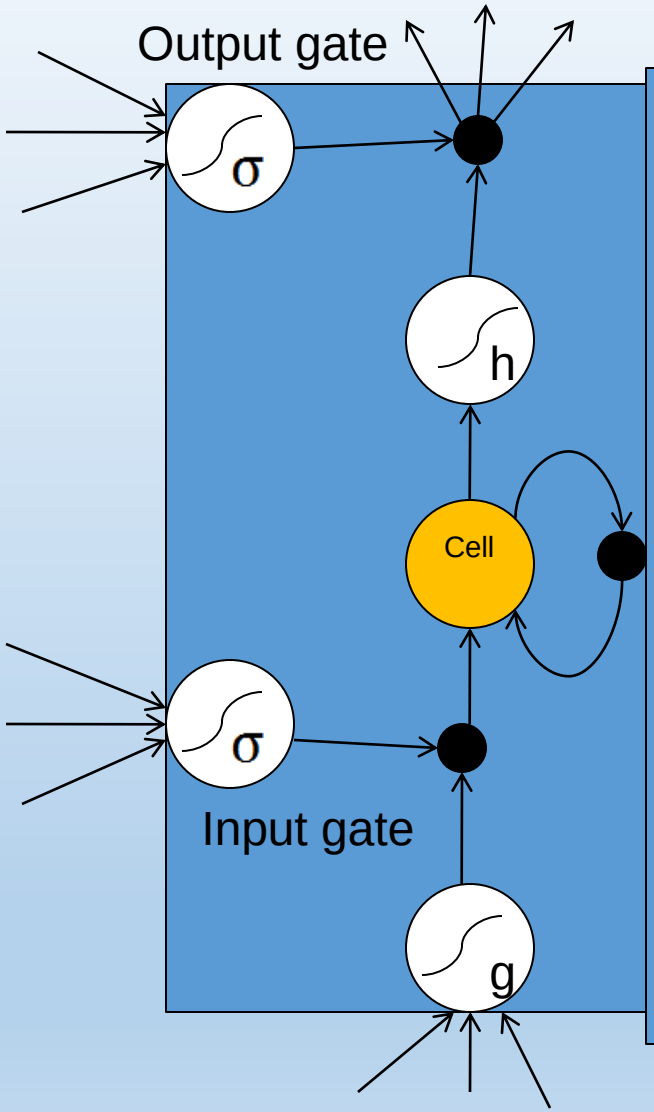
Background — Vector Space Model

Vector space model: An algebraic model for representing text documents (and any objects, in general) as vectors of identifiers, such as, for example, index terms. It is used in information filtering, information retrieval, indexing and relevancy rankings.

Term-document matrix: A mathematical matrix that describes the frequency of terms that occur in a collection of documents.



Background — LSTM



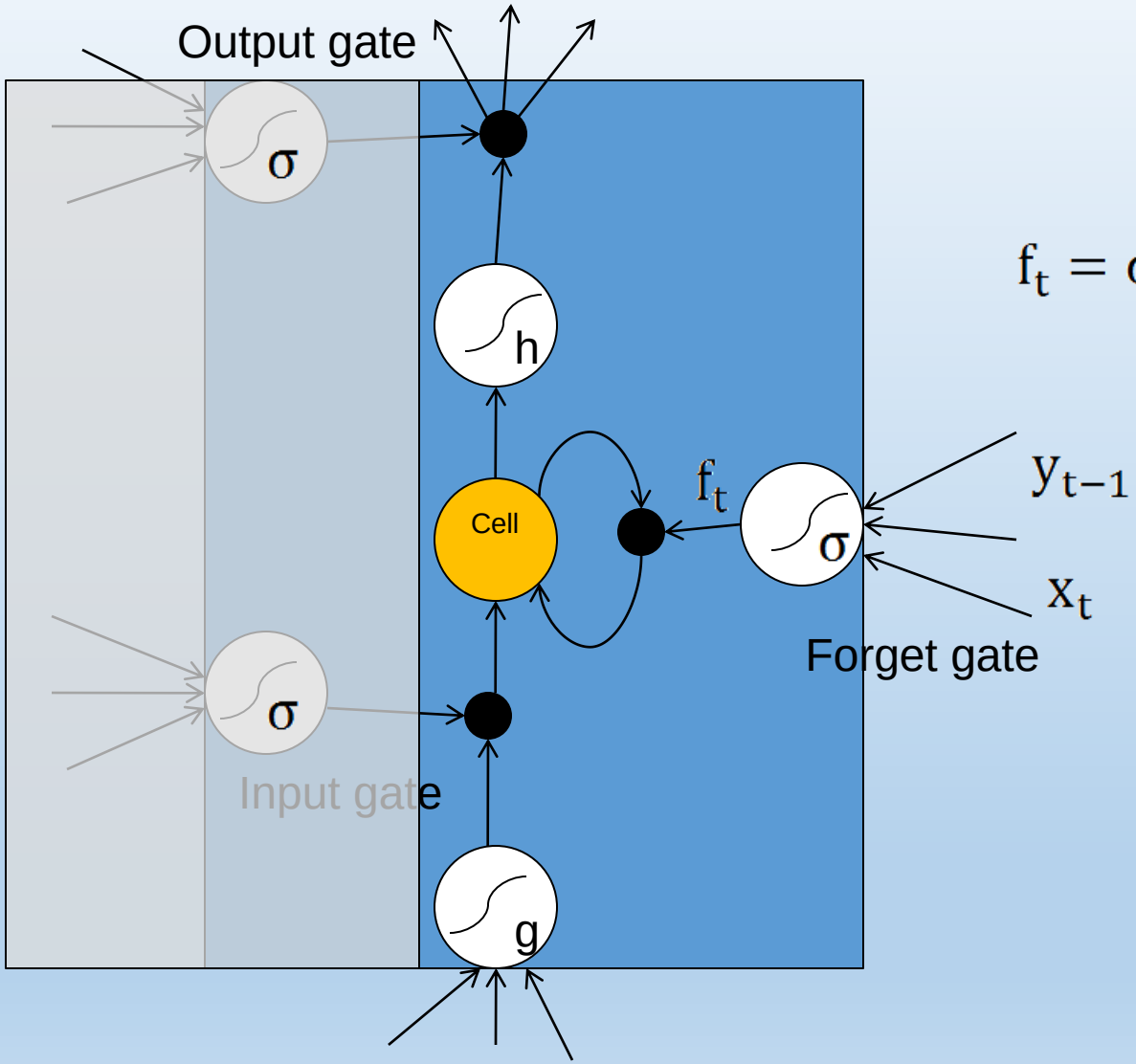
Improved:

1.Forget gate

2. Activation function : logistic sigmoid or tanh.

3. Replace CEC weight 1.0 by the multiplicative forget gate activation .

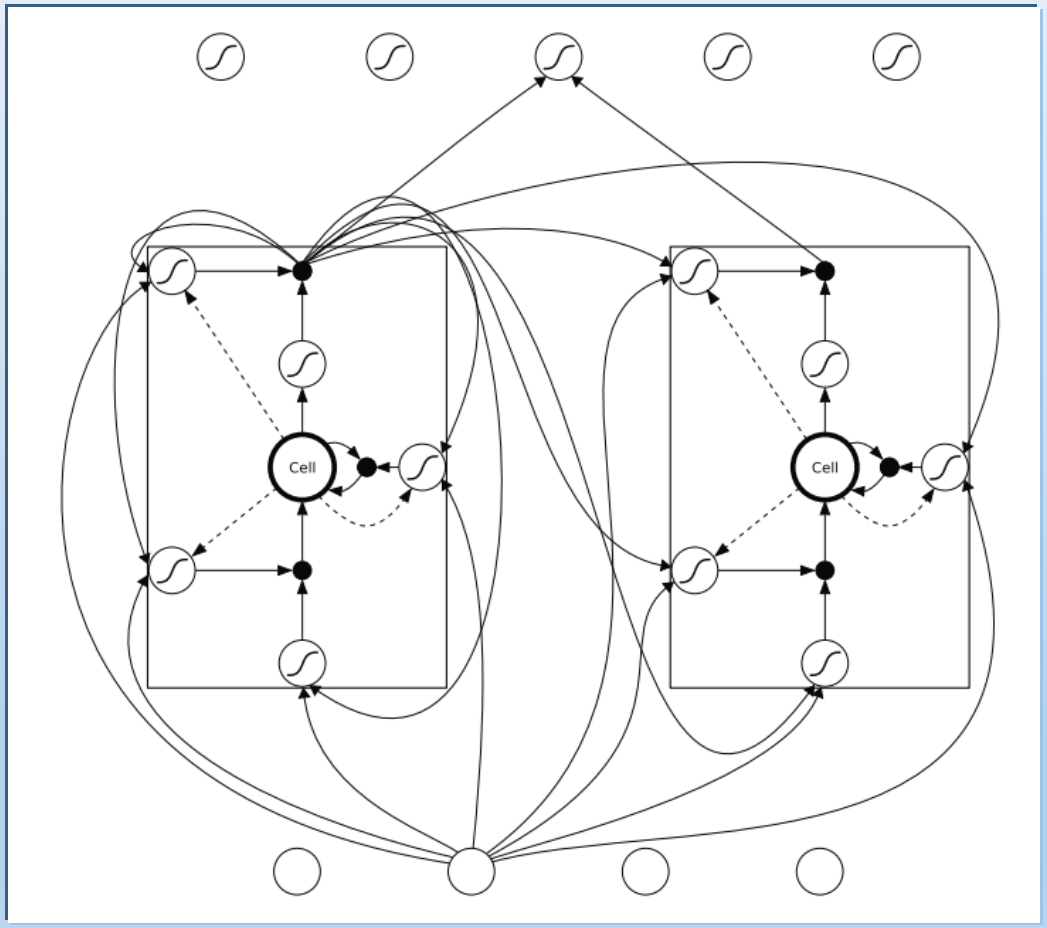
Background — LSTM



$$f_t = \sigma(w_f . [y_{t-1}, x_t] + b_f)$$

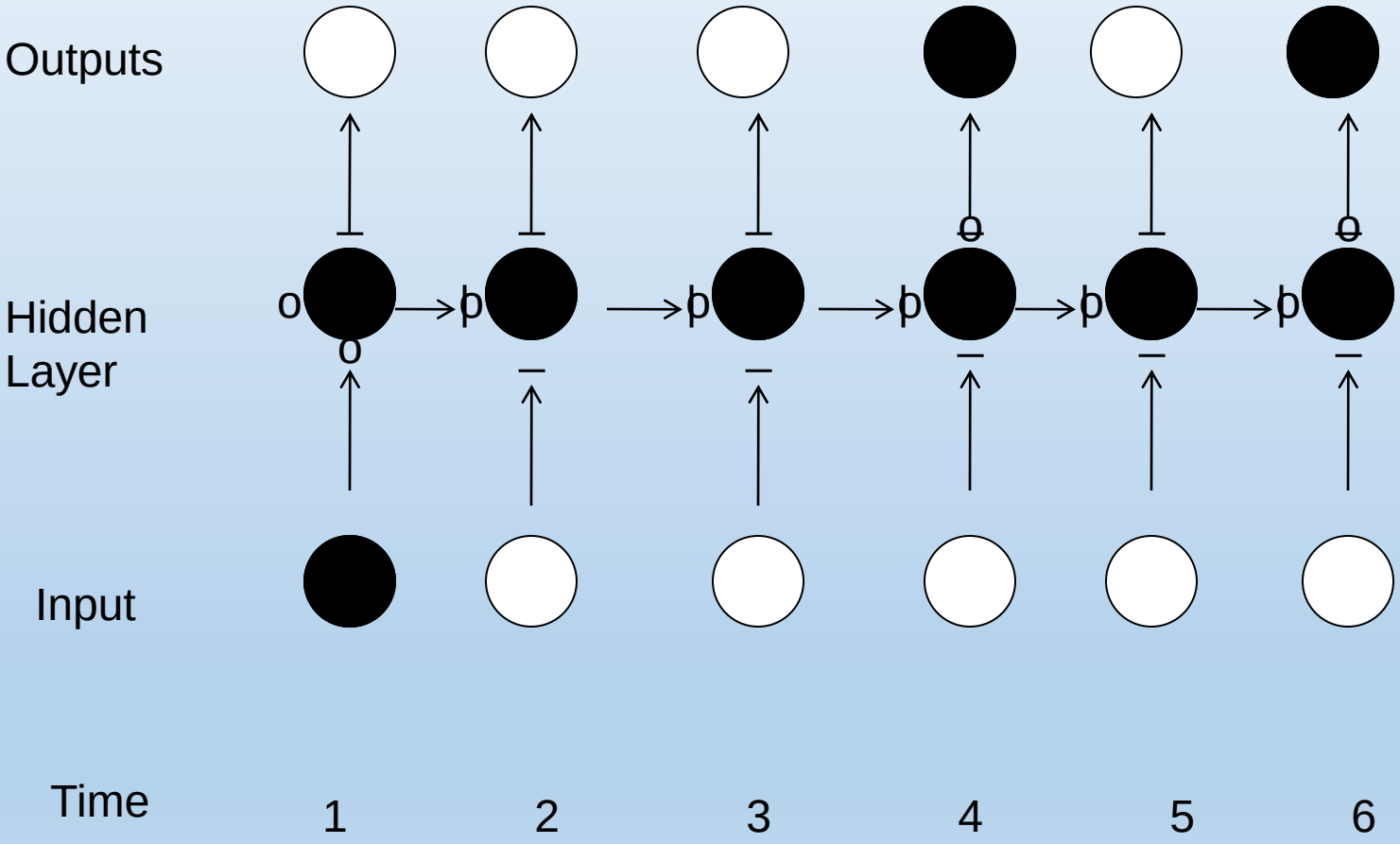
Background — LSTM

Example: An LSTM Network



Background — LSTM

Preservation of gradient information by LSTM



Methods & Result — XML annotation

Software: Notepad++

```
<transcript>
    <question>Alors pour commencer est ce que tu pourrais me présenter ta situation familiale durant l'enfance?</question>
    <answer>(tousse ) euh oui euh
        <father>mon père travaillait le soir</father>
        <mother>donc j'étais surtout élevée par maman</mother>
        <father>j'voyais mon père surtout la fin de semaine</father>
        j'ai rarement été en contact avec mon frère vu qu'il a six ans de plus que moi donc, on fréquentait pas la même éco
        on a déménagé quand j'avais dix ans j'peux juger quand même que j'ai eu une enfance quand même bien,
        <mother>j'ai toujours eu un peu de conflit avant avec ma mère mais euh,</mother>
        <father>ça c'est toujours très bien passé avec mon père.</father>
    </answer>
    <question>OK pis donc tes parents sont pas divorcés?</question>
    <answer>Non mes parents sont toujours mariés ensemble.</answer>
    <question>OK pis euh donc t'es la plus jeune?</question>
    <answer>Ouais</answer>
    <question>Pis euh, de façon générale quel genre de relation t'avais avec ton frère quand t'étais p'tite?</question>
    <answer>Euh j'dirais pas mal inexistante (rire gêné) mon frère a toujours eu une aversion envers moi euh parc'qu'ma mèr
        <mother>je dois l'avouer maman elle ne cachait pas ses pensées, elle m'a préféré à mon frère parce que j'étais plus
    mon frère ayant des difficultés d'apprentissage le voyait, me voyait un peu comme une compétition mais... moi j'l'aimai
    </answer>
    <question>OK et euh, on en a parlé un p'tit peu tantôt mais quel genre de relation t'avais avec ta mère durant ton enfa
    <answer>
        <mother>Avec maman c'était un peu tendu j'me souviens toujours pendant l'été euh, on avait beaucoup de difficultés
    </answer>
    <question>D'accord pis est ce que tu as un exemple pour qualifier ta relation avec ta mère?</question>
    <answer>
        <mother>Hum… (silence) c'est un peu difficile parc'que c'est plus un sentiment intérieur, j'ai jamais vraiment parl
    </answer>
    <question>OK, pis quel genre de relation t'avais avec ton père durant ton enfance?</question>
```

Methods & Result — Word frequency analysis

Method: TF-IDF (term frequency times inverse document frequency)
The term weight of each word represents the importance of the word for the article

```
###TfidfVectorizer can combine CountVectorizer and TfidfTransformer to generate tfidf value directly.
corpus_father = [father.text for father in root.iter('father')]
tfidf_vec_father = TfidfVectorizer()
tfidf_matrix_count_father = tfidf_vec_father.fit_transform(corpus_father)
print(tfidf_vec_father.get_feature_names())
print(tfidf_vec_father.vocabulary_)
print(tfidf_matrix_count_father.toarray())
```

```
['10', 'activités', 'age', 'ai', 'aide', 'aime', 'aimé', 'aimée', 'ait', 'allais', 'aller', 'ami', 'amis', 'an', 'année', 'ans', 'apprécié',
'après', 'arrivait', 'arrêtais', 'asseoir', 'assez', 'assuraient', 'attentionnés', 'aussi', 'autre', 'autrement', 'autres', 'avais', 'avai
t', 'avant', 'avec', 'avoir', 'bah', 'banlieue', 'bas', 'beaucoup', 'bien', 'bin', 'bon', 'bonne', 'bras', 'bénéfique', 'calme', 'caractère
s', 'cause', 'ce', 'centre', 'changement', 'changements', 'chevet', 'chez', 'chicaner', 'choc', 'chose', 'comme', 'complicité', 'comprenne',
'concentrés', 'confier', 'coucher', 'couches', 'couché', 'crème', 'culturel', 'dans', 'de', 'demi', 'des', 'difficultés', 'dirais', 'dire',
'disait', 'discuter', 'dispute', 'disputes', 'dit', 'donc', 'doudous', 'doux', 'droit', 'du', 'déménagement', 'déménagé', 'dérangeais', 'e
n', 'endormais', 'endormir', 'enfants', 'ennuyais', 'ensemble', 'entendre', 'espire', 'essayait', 'est', 'et', 'eu', 'euh', 'explicites', 'e
xpliquait', 'faire', 'faisais', 'fait', 'faite', 'films', 'fin', 'fini', 'finissait', 'fois', 'frère', 'fréquent', 'gaie', 'glacée', 'gorg
e', 'gouts', 'grand', 'gros', 'général', 'géographique', 'haut', 'heu', 'heure', 'hiver', 'hum', 'il', 'ils', 'important', 'importe', 'inqui
ets', 'je', 'jeune', 'jm', 'jouais', 'journée', 'just', 'juste', 'la', 'le', 'les', 'levais', 'lit', 'lolo', 'lui', 'là', 'ma', 'mais', 'mai
son', 'mal', 'mander', 'manipuler', 'manquait', 'matin', 'me', 'mentalité', 'mes', 'mettait', 'milieu', 'minuit', 'minute', 'minutes', 'mo
i', 'moments', 'mon', 'monde', 'monter', 'mère', 'même', 'nait', 'nan', 'neuve', 'niveau', 'non', 'nouvelles', 'nuit', 'obéir', 'obéissanc
e', 'on', 'ont', 'onze', 'ou', 'oui', 'où', 'paisible', 'paix', 'par', 'parc', 'parce', 'parent', 'parents', 'parlait', 'parti', 'partie',
'partir', 'partit', 'pas', 'passait', 'passé', 'pendant', 'permanent', 'peu', 'pile', 'pis', 'plaisant', 'plaisantes', 'plaisir', 'plus', 'p
lutôt', 'politique', 'pour', 'pourrais', 'pouvais', 'presque', 'problèmes', 'proche', 'puis', 'punitions', 'père', 'qu', 'quand', 'que', 'qu
elqu', 'quelque', 'quequ', 'qui', 'quoi', 'racle', 'ram', 'ramasser', 'rappelle', 'rapport', 'rare', 'relation', 'religions', 'remplacé', 'r
enfermée', 'respirer', 'rire', 'rires', 'réglé', 'sais', 'salon', 'savais', 'se', 'semaine', 'sentais', 'ses', 'si', 'similaires', 'soir',
  [[0.         0.         0.         ... 0.         0.         0.        ]
   [0.         0.         0.         ... 0.         0.         0.        ]
   [0.         0.         0.         ... 0.         0.         0.        ]
   ...
   [0.         0.         0.         ... 0.         0.07552874 0.10443793]
   [0.07750755 0.         0.         ... 0.05087484 0.04390082 0.12140837]
   [0.         0.         0.         ... 0.         0.36492667 0.        ]]
```

Methods & Result — Emotion analysis

Method: Emotional matching analysis

```
]:  1 from prettytable import *
    2 table = PrettyTable(['Person_ID','Score_J','Score_A','Score_F','Score_S','Average_J','Average_A','Average_F','Av
    3 for i in range(np.array(score_anger).shape[0]):
    4     table.add_row([Paths[i],score_joy[i][1],score_anger[i][1],score_fear[i][1],score_sadness[i][1],score_joy[i][
    5 print(table)
    6
```

Person_ID	Score_J	Score_A	Score_F	Score_S	Average_J	Average_A	Average_F	Average_S
./277.txt	97.52	66.24	97.14	111.89	0.51	0.54	0.54	0.46
./278.txt	114.36	42.79	54.48	67.94	0.52	0.47	0.44	0.44
./71.txt	74.22	26.02	37.79	53.92	0.52	0.46	0.47	0.42
./527.txt	77.38	62.3	64.61	71.57	0.51	0.51	0.48	0.46
./118.txt	93.75	68.77	86.62	114.21	0.48	0.5	0.48	0.42
./93.txt	95.96	55.47	66.53	109.25	0.5	0.47	0.45	0.45
./90.txt	252.86	72.35	84.41	106.95	0.59	0.51	0.48	0.47
./747.txt	146.53	39.88	81.37	114.97	0.53	0.47	0.48	0.49
./454.txt	88.92	23.52	49.58	52.68	0.56	0.49	0.51	0.41
./116.txt	190.74	118.28	114.78	176.33	0.53	0.52	0.43	0.45
./524.txt	50.97	19.22	41.09	44.79	0.49	0.46	0.48	0.47
./67.txt	123.13	42.37	75.12	98.81	0.49	0.44	0.47	0.42
./7.txt	132.71	70.87	111.05	100.69	0.59	0.57	0.53	0.48
./175.txt	251.1	177.69	255.47	301.25	0.5	0.5	0.46	0.47
./739.txt	95.97	54.0	80.45	107.89	0.5	0.46	0.45	0.48
./15.txt	228.11	89.0	120.47	209.47	0.53	0.49	0.47	0.44
./72.txt	188.17	87.72	135.12	168.53	0.51	0.5	0.51	0.45

Methods & Result — LSTM

```
1 train(net, context, epochs)
```

```
[Epoch 0 Batch 100/779] elapsed 276.19 s, avg loss 0.002344, throughput 2.92K wps
[Epoch 0 Batch 200/779] elapsed 301.65 s, avg loss 0.001810, throughput 2.58K wps
[Epoch 0 Batch 300/779] elapsed 348.65 s, avg loss 0.001432, throughput 2.45K wps
[Epoch 0 Batch 400/779] elapsed 348.03 s, avg loss 0.001347, throughput 2.29K wps
[Epoch 0 Batch 500/779] elapsed 330.46 s, avg loss 0.001399, throughput 2.24K wps
[Epoch 0 Batch 600/779] elapsed 320.99 s, avg loss 0.001170, throughput 2.43K wps
[Epoch 0 Batch 700/779] elapsed 288.23 s, avg loss 0.001230, throughput 2.67K wps
Begin Testing...
[Batch 100/782] elapsed 354.15 s
[Batch 200/782] elapsed 352.49 s
[Batch 300/782] elapsed 388.84 s
[Batch 400/782] elapsed 551.22 s
[Batch 500/782] elapsed 451.92 s
[Batch 600/782] elapsed 471.41 s
[Batch 700/782] elapsed 467.54 s
[Epoch 0] train avg loss 0.001495, test acc 0.86, test avg loss 0.309759, throughput 2.50K wps
```


Thanks