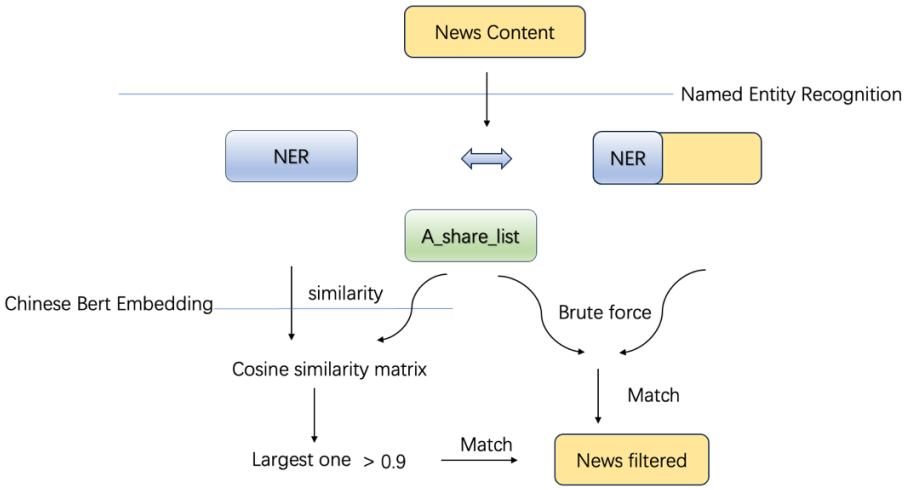
Task1 Q1

- Conclusive Strategy



I leveraged LAC, a Baidu NLP tool known for its proficiency in named entity recognition (NER), accessible at https://github.com/baidu/lac.

Following this process, there remain 889,496 news with approximately 12,000 filtered out due to the absence of named entities.

Subsequently, I employed BERT Chinese to transform the 'a_list_share' into a 'bert_vector_dict.' Utilizing cosine similarity, I computed a similarity matrix for each Named Entity Recognition (NER) and identified the highest similarity score. To determine the appropriate threshold, I conducted tests with various NER and company names, settling on 0.9. If the calculated similarity surpasses 0.9, it is considered a match, while scores below 0.9 are deemed non-matching.

After implementing these steps, the dataset has been narrowed down to 441,802 news articles. However, it came to my attention that certain company names were not identified by the NER, making the similarity method ineffective in those cases. To address this, I employed a brute force method to determine if the A-share list company name was mentioned in the news. Following this additional step, the dataset was further refined to 537,586 news.

- Initial Trials

Prior to these refined methods, I made preliminary attempts, which, although deemed less effective, were documented for record-keeping purposes.

I attempted to use Jieba for word segmentation but encountered an issue where it segmented '中石油' into '中' and '石油.' Additionally, I explored other tools such as KeyBert; however, I

observed that the key terms identified were not often the company names. I ultimately settled on using Named Entity Recognition (NER).

Subsequently, I faced another challenge — selecting an appropriate similarity metric. Prior to employing Bert embeddings, I experimented with alternatives like TF-IDF. However, this approach proved to be stupid, given that I had already extracted Named Entities (NER), rendering IDF unsuitable. While TF similarity exhibited promise, I encountered issues when utilizing unigrams to construct the matrix, as individual tokens led to mismatches. To address this, I incorporated both unigrams and bigrams. Despite these efforts, the rules I implemented were still insufficient to cover all news content in my testing. The following codes represent my attempts at engineering and subsequent testing.

```
    # 将字中间加入空格
    s1, s2 = add_space(s1), add_space(s2)
    # 转化为TF矩阵
    #cv = CountVectorizer(tokenizer=lambda s: [s[i:i+2] for i in range(len(s) - 1)] + s.split())

    cv = CountVectorizer(tokenizer=lambda s: [s[i:i+2] for i in range(len(s) - 1)])
    #cv = CountVectorizer(tokenizer=lambda s: s.split())

    corpus = [s1, s2]
    vectors = cv.fit_transform(corpus).toarray()
    # 计算TF系数
    return np.dot(vectors[0], vectors[1]) / (norm(vectors[0]) * norm(vectors[1]))

output = []  # 创建一个空列表

for word in df['NamedEntityRecognition'][7]:
    #findname = False
    for index, row in a_share_df.iterrows():
        similarity_name = tf_similarity(word, row['name'])
        similarity_fullname = tf_similarity(word, row['fullname'])

        #if  similarity_name > 0.55 and similarity_fullname > 0.5:
        if similarity_name > 0.9 or similarity_fullname > 0.65: #and
            output.append(row['fullname'])
            #findname = True
            print("识别主体:", word)
            print("匹配项",row['name'],row['fullname'])
            print(similarity_name,similarity_fullname)
            #break
    #if findname:
        #continue

#print(output)
```

```
识别主体: 冠农股份
匹配项 冠农股份 新疆冠农果茸股份有限公司
1.0000000000000002 0.5222329678670935
识别主体: 南风化工公司
匹配项 ST南风 南风化工集团股份有限公司
0.25819888974716115 0.674199862463242
识别主体: 山西钾肥有限公司
匹配项 西山煤电 山西西山煤电股份有限公司
0.2182178902359924 0.656599153958537
```

```
[ ]  df10['Explicit_Company']

0    [中国能源建设股份有限公司, 中国建设银行股份有限公司]
1                                    []
2                    [中国国际航空股份有限公司]
3                      [山东胜利股份有限公司]
4                                    []
5                                    []
6                                    []
7    [新疆冠农果茸股份有限公司, 南风化工集团股份有限公司]
8                    [山东博汇纸业股份有限公司]
9        [申能股份有限公司, 中国国际金融股份有限公司]
Name: Explicit_Company, dtype: object
```

Task1 Q2

I'd like to use an existing Chinese sentiment analysis library. In order to evaluate the accuracy of existing Chinese sentiment analysis libraries, I employed the following news articles for testing purposes.

text=' 受美国股市上扬以及未来数日一系列 IPO 活动提振，恒指昨日盘中重上 19000 点，收报 18960.48 点，升 179.55 点，成交 467.67 亿港元；国企指数升 144.34 点，收 8546.6 点。蓝筹方面，中移动、汇控、和黄分别涨 1.71%、0.49%和 1.04%；地产股长实、信置、恒地、新地及恒隆地产升 0.57%至 2.25%不等，新世界发展倒跌 0.14%。银行股恒生、东亚、中银香港升 0.29%、0.6%和 1.19%。内地电讯股网通、联通和非蓝筹的中电信各升 0.13%、0.34%和 2.87%。 中资金融股普遍继续升高。建行、招行、工行、中行及交行升 0.56%至 1.63%。中资保险股中，国寿、平保、财险各升 1.39%、2.4%及 1.67%。二线金融股中信国金、光控、闽信、中保均有不同涨幅。中资房产股重拾升势，新世界中国及中国海外均涨逾 5%，华润置地、合生创展、越投升幅介乎 2.62%至 3.4%；瑞房斥 5.82 亿元人民币收购上海地皮，升 1.66%；不过，计划分拆附属公司上市的上海置业回吐 0.82%。资源股方面，表现最强的是钢铁股。外电报道中国最大钢铁公司宝钢母公司可能向新日本制铁或韩国浦项综合制铁公司出售股权，这表明外资对中国钢铁股需求旺盛，马钢、鞍钢同创新高，分别大涨 6.925%和 7.44%。油价持续上扬，中石油盘中创 10.02 元新高，收升 3.125%，下周“染蓝”的中石化亦涨 3.54%。周二大跌后，市场普遍认为恒指仍将上试 19000 点，但这一点位将成为短期阻力位。券商普遍唱好内地经济发展，中银国际预期，中国 2006 年 GDP 增长可达 10.5%，上升周期可持续至 07 年甚至 08 年。富昌证券蔺常念称，股市重新恢复乐观气氛，外围股市走高，港股没有理由不涨，在人民币升值题材下，投资者会继续买入中资金融股和电讯股。凯基证券首席营运长邝民彬称，市场继续跟随华尔街和日本股市涨势反弹，因美政府公布三季度 GDP 数据显示，经济增长情况好于预期，减轻了市场忧虑情绪。工商东亚称，市场连续两天反弹表明投资者仍信心十足，预计恒指年底前将重新上试历史高位 19404 点。法兴证券 Andrew Clarke 表示，一些欧美基金把周二股市下挫看作入市好机会，但港股短期内再升 10%的看法相当乐观，毕竟今年以来已经上扬了 27.5%；股市继续上涨将促使基金兑现利润、提前封账。'

I think it's definitely positive. And the confidence should be above 0.5.

- The first one is jiagu. https://github.com/ownthink/Jiagu

```
[27] import jiagu

     text = '受美国股市上扬以及未来数日一系列IPO活动提振，恒指昨日盘中重上19000点，收报18960.48点，升179.55点，成交467.67亿港元
     sentiment = jiagu.sentiment(text)
     print(sentiment)

     ('negative', 1.0)
```

The outcome is markedly unfavorable, with a confidence level of 1 indicating a high certainty of negativity

- The second one is baidu sentiment analysis API. It has three labels. Also, it labelled the text sentiment 0, negative probability 0.591, positive probability 0.409, which is unacceptable.

```python
import requests
import json

url = "https://aip.baidubce.com/rpc/2.0/nlp/v1/sentiment_classify?charset=UTF-8&access_token=24.65f8f8f92be545a2

headers = {
    "Content-Type": "application/json"
}

data = {
    "text": "受美国股市上扬以及未来数日一系列IPO活动提振，恒指昨日盘中重上19000点，收报18960.48点，升179.55点，成交467.67亿港元
}

response = requests.post(url, headers=headers, data=json.dumps(data))

# Print the response
print(response.json())
```

```
0.0922065, 'negative_prob': 0.591493, 'positive_prob': 0.408507, 'sentiment': 0}], 'log_id': 1725358009973991631}
```

- The third one and also the last one is bixin. https://github.com/bung87/bixin. The sentiment analysis yields a positive result, with a commendable score of 0.48. This outcome is deemed highly satisfactory.

```
text ="受美国股市上扬以及未来数日一系列IPO活动提振，恒指昨日盘中重上19000点，收报18960.48点，升179.55点，成交467.67亿港元；国企指数升144.34点，收8546.6点。
predict(text)
```

```
0.48
```

I hold a positive assessment, having cross-verified with several news articles exhibiting sentiment scores within the range of -0.2 to 0.2, indicating a relatively reasonable outcome. Therefore, I chose this one to do the sentiment analysis. Positive (label=1) if predict(text) > 0, Negative (label=0) if predict(text) < 0.

Task 2 Q3

In my view, edges such as 'compete,' 'cooperate,' 'dispute,' and 'same_industry' are deemed undirected, signifying bidirectional relationships. Conversely, 'invest' and 'supply' are identified as directed edges.