Introduction to Deep Learning Final Project

Group 7

2020313223 Kim Sang Jun

2017310869 Lee Jin Mo

Contents

1

Introduction & Background

1 Introduction & Background

Problem Tackled

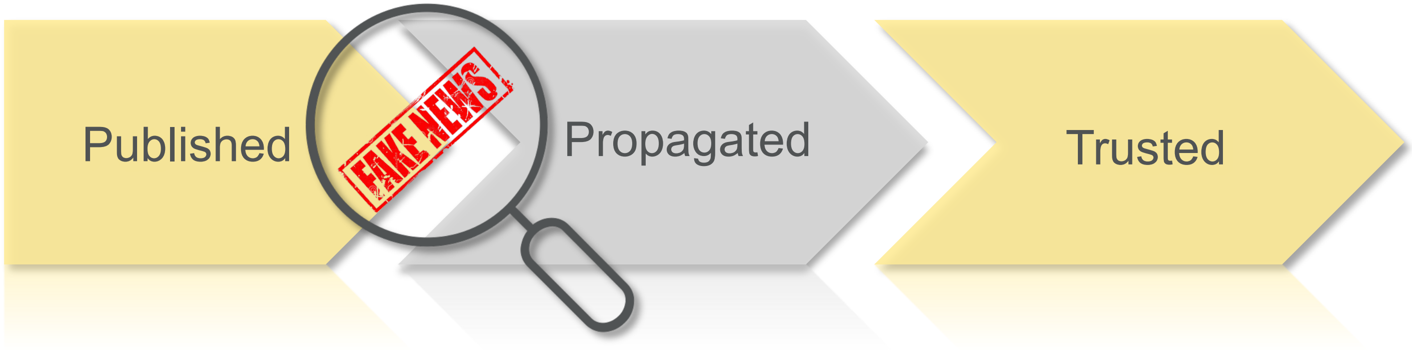
Fake News Detection

“ *Misinformation spreads* ”

faster, farther, deeper, and more widely

Vosoughi et al., 2018

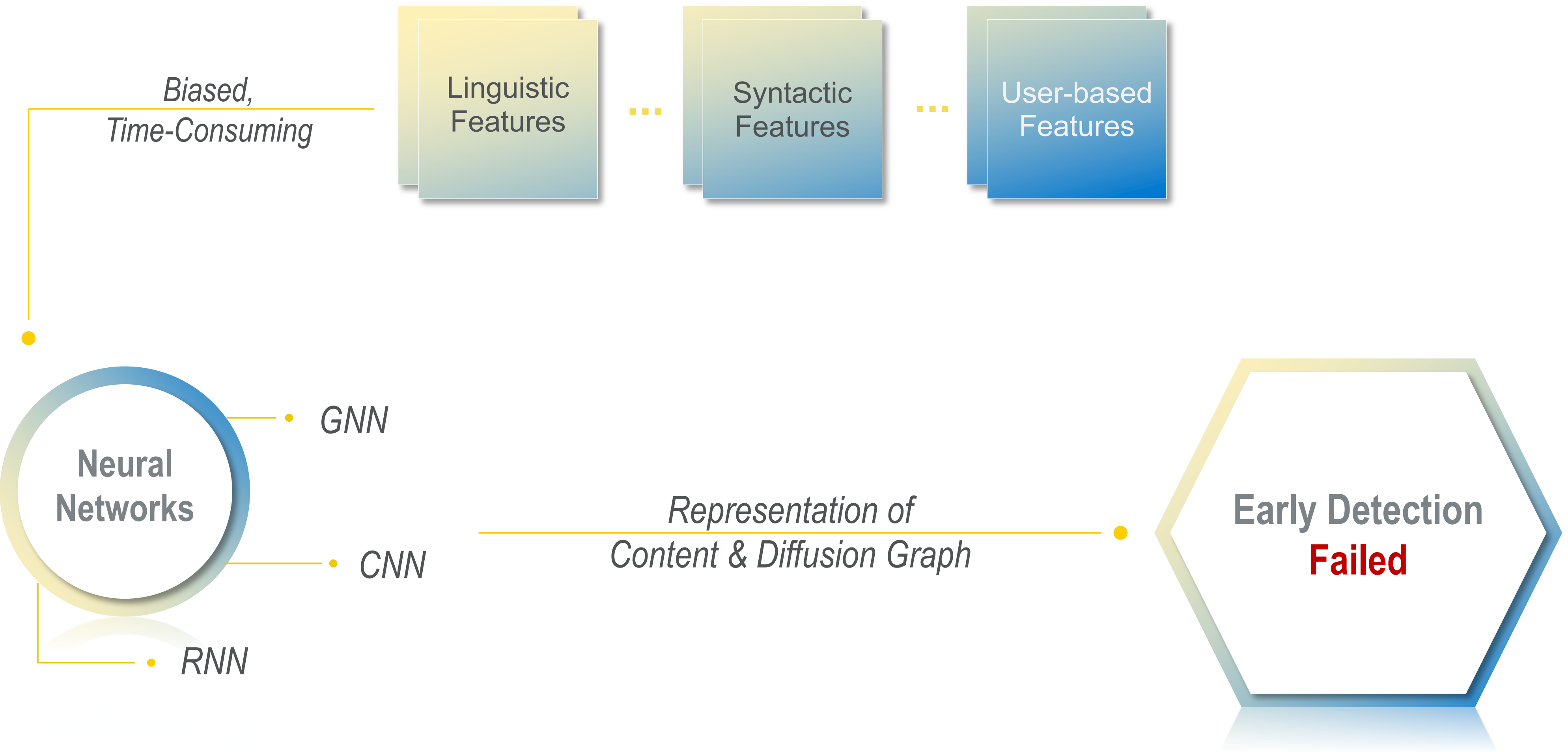
Early Detection



1 Introduction & Background

Previous Challenges

- Feature based detection

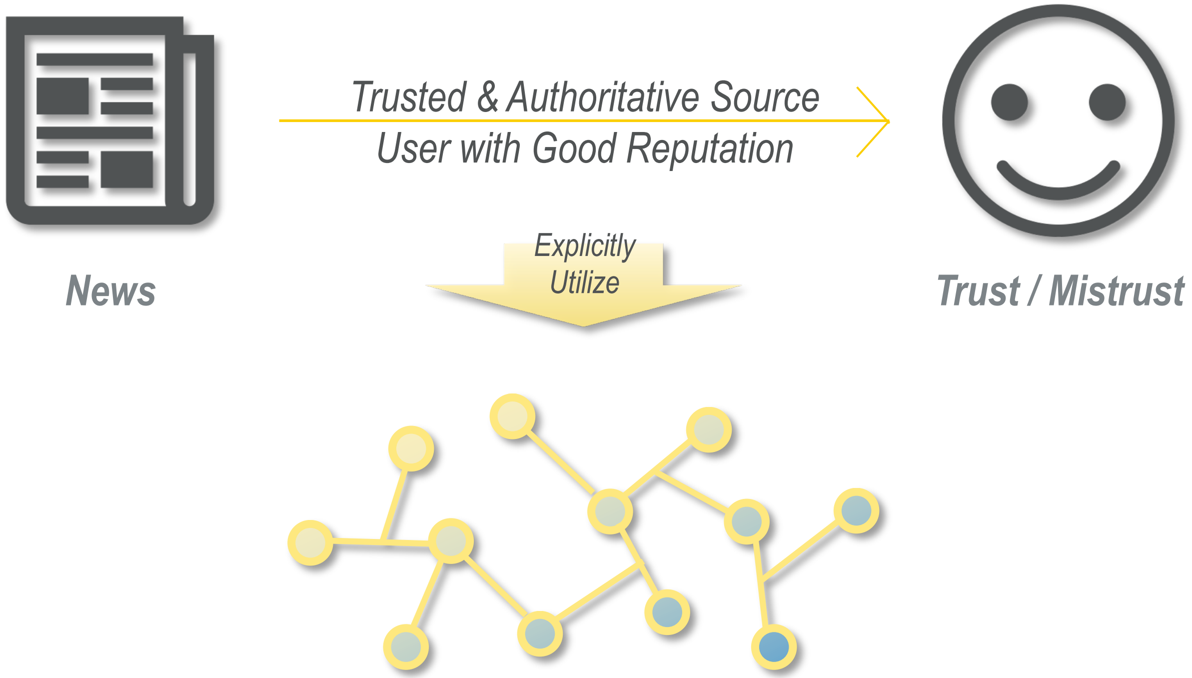


2 Methods

2 Methods

New Approach: Publisher & User Credibility Prediction

Credibility as Supervised Information

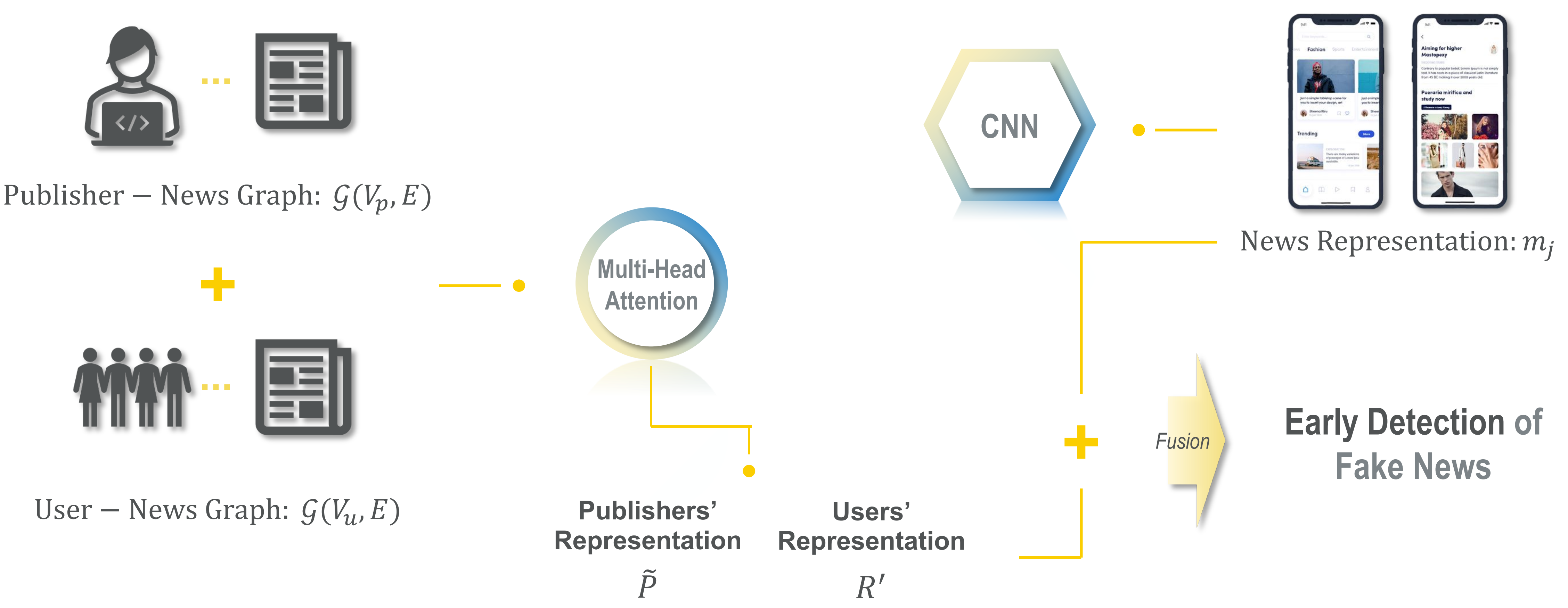


Publisher & User Credibility + News
Heterogeneous Graph

2 Methods

Structure-Aware Multi-head Attention Network

Attention & CNN Combined Structure



2 Methods

Publisher Credibility Prediction

Attention Module to Predict Credibility

Multi-Head Attention

$$Attention(Q, K, K) = Z_h = softmax\left(\frac{QW_hK^T}{\sqrt{d}} \odot (D^p)^{-\frac{1}{2}} A^{pn} (D^n)^{-\frac{1}{2}}\right) K$$

: Publisher Embeddings

: News Embeddings

: Adjacency Matrix

: Credibility Scores _ Unreliable(0) / Uncertain(1) / Reliable(2)

$$p_i\left(c \middle| \mathcal{G}(V_p, E), \mathcal{P}; \theta_1\right) = softmax\left(\tilde{P}_i W_p + b_p\right)$$

$$\tilde{P} = ELU([Z_1; Z_2; \ldots; Z_H]W_o) + P$$

: Publishers' Representations

Same Procedure Applied to User Credibility Prediction

2 Methods

User Credibility Prediction

Attention Module to Predict Credibility

Multi-Head Attention

$$Attention(Q, K, K) = Z_h = softmax\left(\frac{QW_hK^T}{\sqrt{d}} \odot (D^p)^{-\frac{1}{2}} A^{pn} (D^n)^{-\frac{1}{2}}\right) K$$

: User Embeddings

: News Embeddings

: Adjacency Matrix

: Credibility Scores _ Unreliable(0) / Uncertain(1) / Reliable(2)

$$p_{ij}(c|\mathcal{G}(V_u, E), \mathcal{U}; \theta_2) = softmax(\tilde{R}_{ij} W_r + b_r)$$

$$\tilde{R}_j = ELU([Z_1; Z_2; \dots; Z_H]W_o) + R_j$$

: User j's Representations

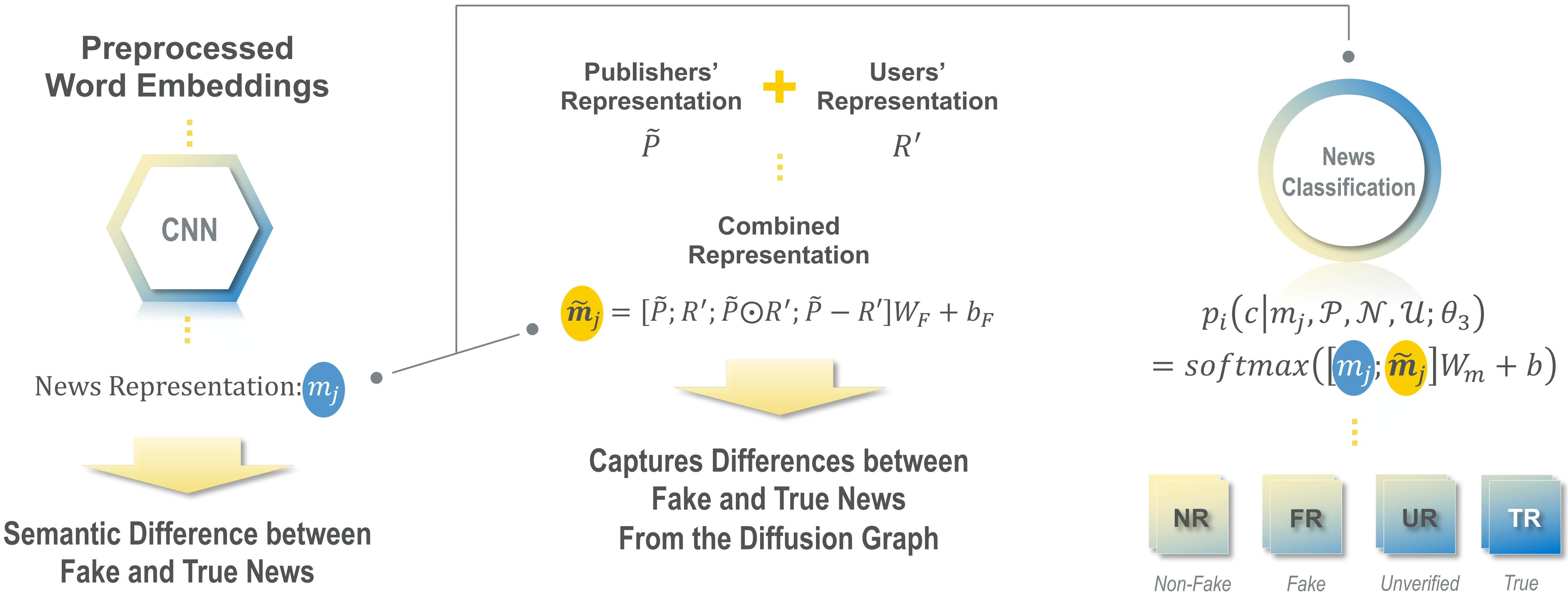
$$R' = \sum_{k=1}^{K} \alpha_k \tilde{R}_k$$

: K different user's representation who had reposted the same news

2 Methods

Fusion Attention Unit

News Representation + Credibility Prediction



2 Methods

Combined Cross Entropy Loss

Optimizing Every Tasks Together

Simultaneously Optimize
Credibility Prediction & Fake News Detection

Cross Entropy Loss

$$\mathcal{L}\left(c \mid \underbrace{\mathcal{G}\left(V_p, E\right)}_{\text{Publisher – News graph}}, \overbrace{\mathcal{G}(V_u, E)}^{\text{User – News Graph}}, \mathcal{N}; \theta\right) = \mathcal{L}_p + \mathcal{L}_u + \mathcal{L}_n$$

- : Objective Function for Publisher Credibility Prediction
- : Objective Function for User Credibility Prediction
- : Objective Function for Fake News Detection

2 Methods

Data Preprocessing

Natural Language Process

Cleaning

Padding

Word2Vec

```
def clean_str_cut(string, task):
    """
    Tokenization/string cleaning for all datasets except for SST.
    Original taken from https://github.com/yoonkim/CNN_sentence/blob/master/process_data.py
    """
    if task != "weibo":
        string = re.sub(r"[^A-Za-z0-9(),!?#@\'\`]", " ", string)
        string = re.sub(r"\'m", " am", string)
        string = re.sub(r"\'s", " \'s", string)
        string = re.sub(r"\'ve", " have", string)
        string = re.sub(r"n\'t", " not", string)
        string = re.sub(r"\'re", " are", string)
        string = re.sub(r"\'d", " had", string)
        string = re.sub(r"\'ll", " will", string)
```

```
def pad_sequence(X, max_len=50):
    X_pad = []
    for doc in X:
        if len(doc) >= max_len:
            doc = doc[:max_len]
        else:
            doc = [0] * (max_len - len(doc)) + doc
        X_pad.append(doc)
    return X_pad
```

	User ID	Publisher ID	News	Class
1	1575	7247…	american family association …	unverified
2	1407	3585…	this week's top story: george wins florida …	false
3	2648	7756…	clinton hides failing health? …	unverified
4	2793	3645…	fukushima: highly radioactive water …	false
…	…	…	…	…

2 Methods

Data Preprocessing

Natural Language Process

Cleaning

Padding

Word2Vec

	User ID	Publisher ID	News	Class
1	1575	7247…	american family association …	unverified
2	1407	3585…	this week's top story: george wins florida …	false
3	2648	7756…	clinton hides failing health? …	unverified
4	2793	3645…	fukushima: highly radioactive water …	false
…	…	…	…	…

Converted into Embedding Vector Suitable for training

```
def vocab_to_word2vec(fname, vocab):
    """
    Load word2vec from Mikolov
    """
    word_vecs = {}
    model = gensim.models.KeyedVectors.load_word2vec_format(fname, binary=True)
    count_missing = 0
    for word in vocab:
        if model.__contains__(word):
            word_vecs[word] = model[word]
        else:
            #add unknown words by generating random word vectors
            count_missing += 1
            word_vecs[word] = np.random.uniform(-0.25, 0.25, w2v_dim)
            # print(word)

    print(str(len(word_vecs) - count_missing)+" words found in word2vec.")
    print(str(count_missing)+" words not found, generated by random.")
    return word_vecs
```

```
def build_vocab_word2vec(sentences, w2v_path='numberbatch-en.txt'):
    """
    Builds a vocabulary mapping from word to index based on the sentences.
    Returns vocabulary mapping and inverse vocabulary mapping.
    """
    # Build vocabulary
    vocabulary_inv = []
    word_counts = Counter(itertools.chain(*sentences))
    # Mapping from index to word
    vocabulary_inv += [x[0] for x in word_counts.most_common() if x[1] >= 2]  #
    # Mapping from word to index
    vocabulary = {x: i for i, x in enumerate(vocabulary_inv)}

    print("embedding_weights generation.......")
    word2vec = vocab_to_word2vec(w2v_path, vocabulary)     #
    embedding_weights = build_word_embedding_weights(word2vec, vocabulary_inv)
    return vocabulary, embedding_weights
```

3

Evaluation & Results

3 Evaluation & Results

Datasets Overview

Datasets

	# news	# non-fake news(NR)	# fake news (FR)	# unverified news (UR)	# true news (TR)	# users	# retweets
Twitter15	1490	374	370	374	372	276,663	331,612
Twitter16	818	205	205	203	205	173,487	204,820
Weibo	4664	2351	2313	0	0	2,746,818	3,805,656

3 Evaluation & Results

Evaluation Metrics

F1-score

		Real Class	
		True	False
Predicted Class	True	**TP**	**FP**
	False	**FN**	**TN**

Precision

$$\frac{TP}{TP + FP}$$

F1-Score

$$2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Recall

$$\frac{TP}{TP + FN}$$

3 Evaluation & Results

Fake News Detection Evaluation

Twitter15

SMAN *Model proposed from the paper*

	Precision	Recall	F1-Score
NR	0.865	0.988	0.922
FR	0.975	0.917	0.945
TR	0.938	0.893	0.915
UR	0.951	0.917	0.933
ACC	0.929		

GLAN *State-of-the-art before SMAN*

	F1-Score
NR	0.924
FR	0.917
TR	0.852
UR	0.927
ACC	0.905

3 Evaluation & Results

Fake News Detection Evaluation

Twitter16

SMAN *Model proposed from the paper*

	Precision	Recall	F1-Score
NR	0.936	0.957	0.946
FR	0.976	0.870	0.920
TR	0.857	0.933	0.894
UR	0.979	0.979	0.979
ACC		0.935	

GLAN *State-of-the-art before SMAN*

	F1-Score
NR	0.921
FR	0.869
TR	0.847
UR	0.968
ACC	0.902

3 Evaluation & Results

Fake News Detection Evaluation

Weibo

SMAN *Model proposed from the paper*

	Precision	Recall	F1-Score
NR	0.967	0.936	0.951
FR	0.937	0.967	0.952
ACC	0.951		

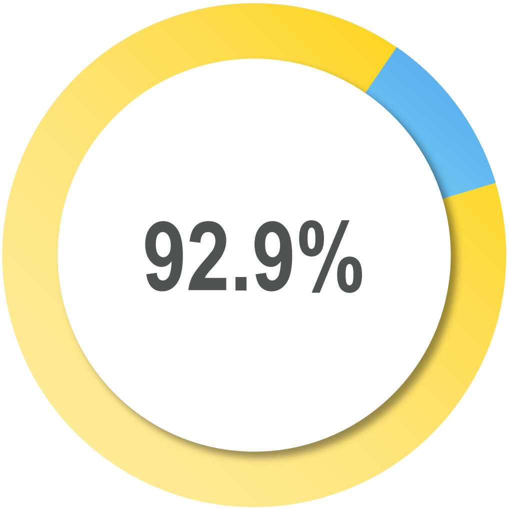
GLAN *State-of-the-art before SMAN*

	F1-Score
NR	0.946
FR	0.945
ACC	0.946

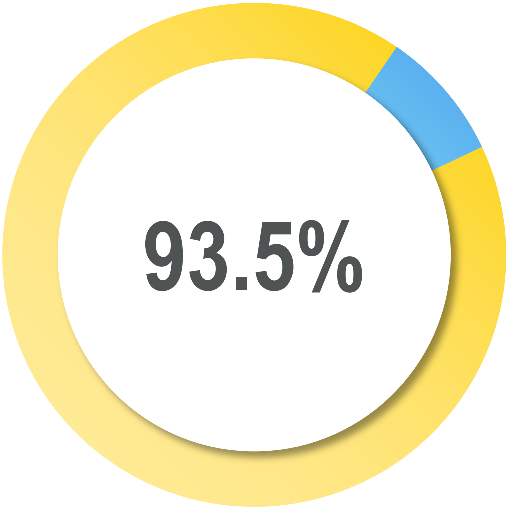
3 Evaluation & Results

Fake News Detection Evaluation

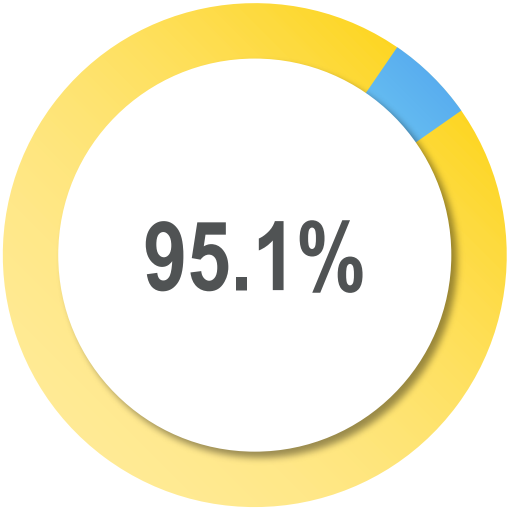
Accuracy Comparison among Datasets



Twitter 15



Twitter 16



Weibo

3 Evaluation & Results

Credibility Prediction Validity

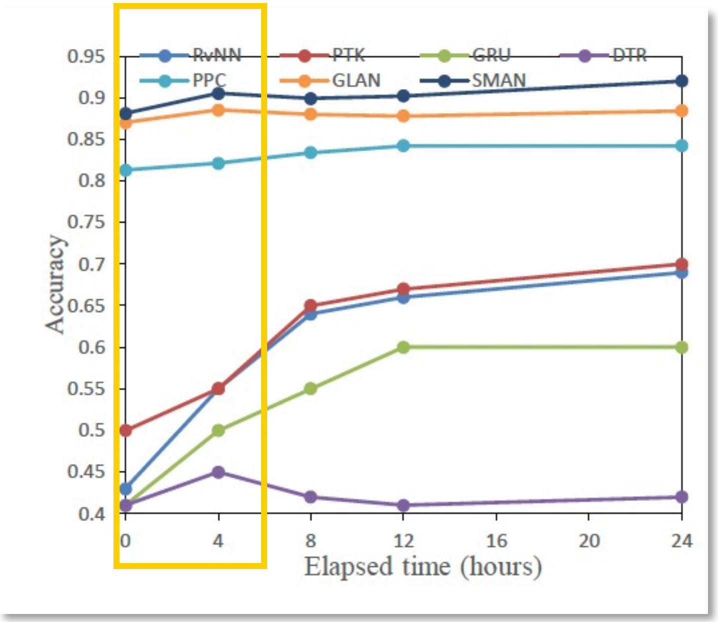
Ablation Study Result

Models	Twitter15 Accuracy	Twitter16 Accuracy	Weibo Accuracy
SMAN w/ Publisher & User Credibility	0.929	0.935	0.951
SMAN w/o Publisher Credibility	0.887	0.913	0.930
SMAN w/o User Credibility	0.905	0.880	0.938
SMAN w/o Publisher & User Credibility	0.863	0.851	0.911

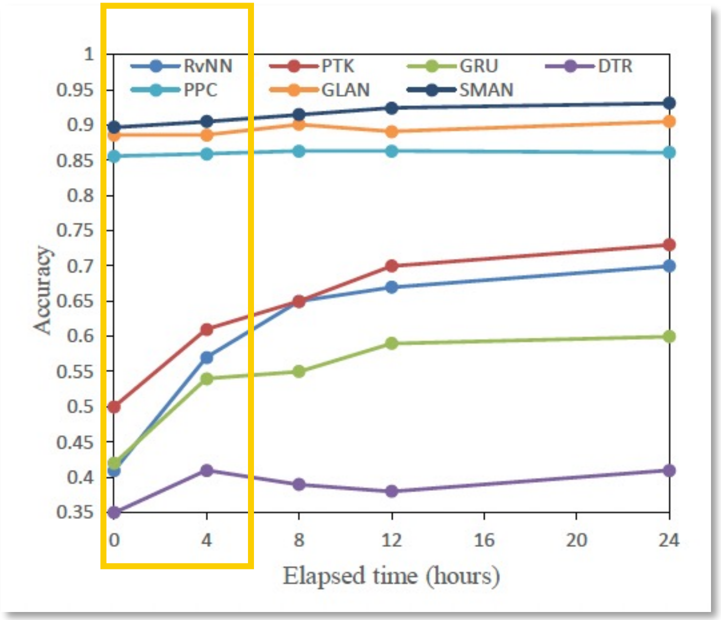
3 Evaluation & Results

Early Detection Evaluation

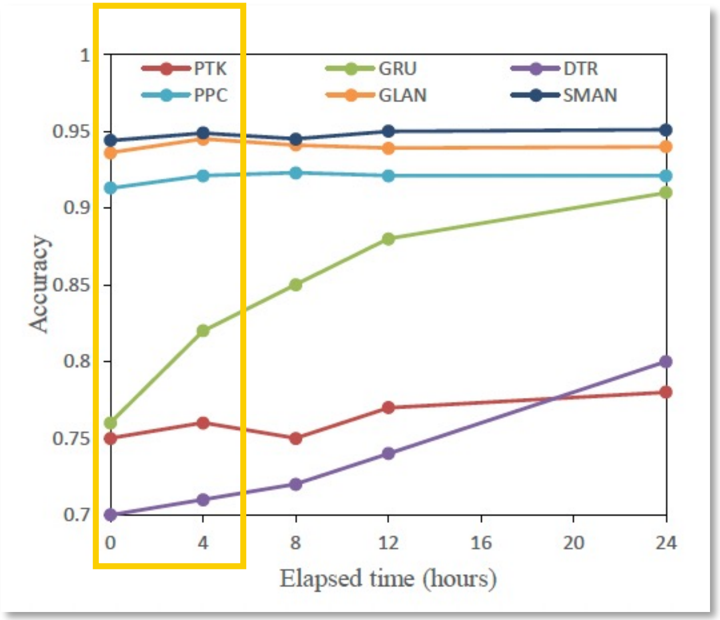
Comparison between Previous Studies



Twitter 15



Twitter 16



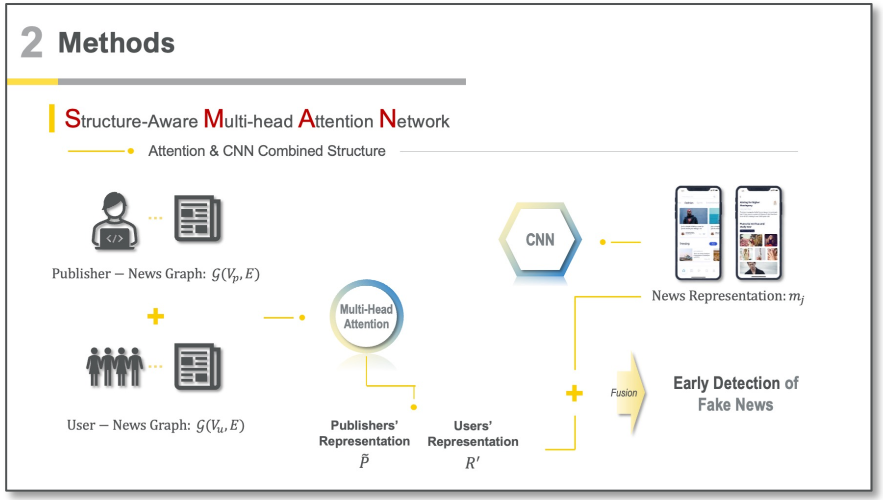
Weibo

4 Challenges

4 Challenges

Concept of the paper

- Several tasks going on simultaneously



Overall Concept of the paper itself was unfamiliar

It was not about only implementing one method to one task,

but rather implement many methods to many tasks simultaneously.

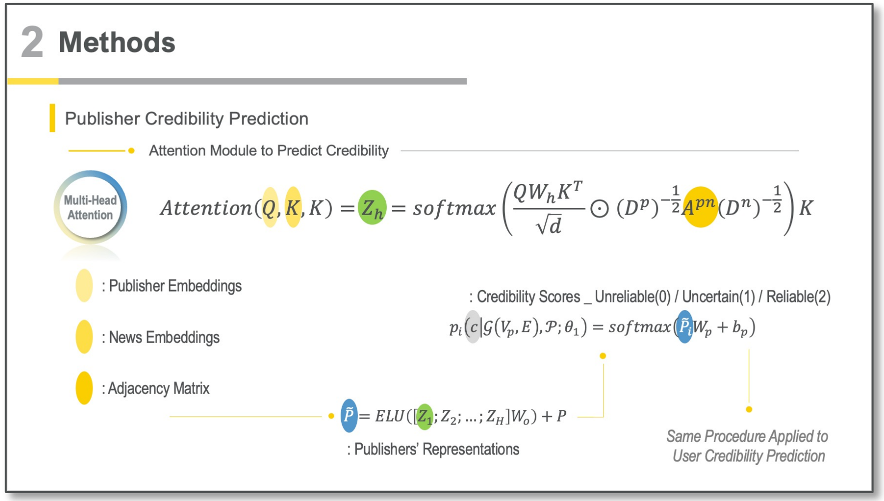
Therefore, we had to go through previous studies in order to get

knowledge about the domain and methodologies for this problem.

4 Challenges

Mathematical Structure

Multi-head Attention Module



It was necessary to understand mathematical structure of the model

in order to best explain the whole paper

Multi-head attention was also an unfamiliar model to us at first.

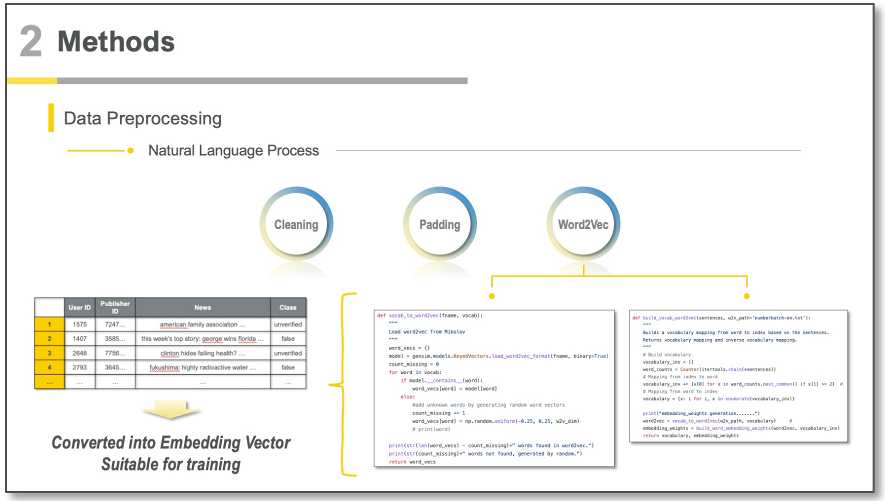
Therefore, we had to re-read the paper several times and

conduct additional research about the model and its mathematical formulas.

4 Challenges

Pre-processed Data

- Data given were already pre-processed



Data were all already pre-processed before given to us

Even the embedded vectors were not able to modify

Basic pre-processing steps were provided, but it was very unfriendly.

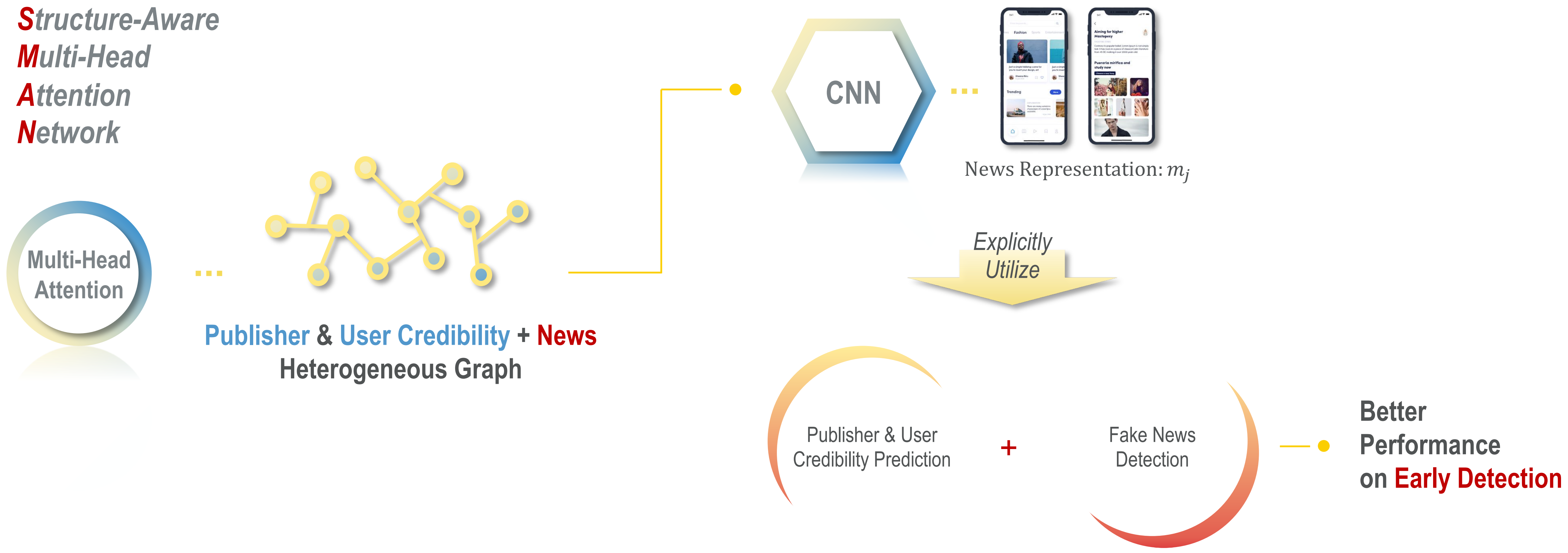
Therefore, it was nearly impossible to check out the dataset, and get any insights necessary to understand the performance of the model

5 Conclusion

5 Conclusion

Key Points Revisiting

Fake News Detection with Publisher & User Credibility Prediction



Introduction to Deep Learning
Final Project

Thank You !