Department of Computer Engineering,
College of Technology, G.B. Pant University of Agriculture and Technology

Paraphrasing Detection Using Dependency Tree Recursive Autoencoder

Deepak Singh Rana
M.Tech. student (ID : 49639)

- **Paraphrase** means express same meaning using different words.
  - E.g.:
    [people like beautiful things.]
    [beautiful things are liked by every one.]

- **Paraphrasing detection** is a process of detection whether two sentence are semantically (by meaning) similar or not.

- Paraphrasing detection plays an important role in problem like question answering, automatic summarization, information retrieval etc.

- Paraphrasing detection is divided into two sub problem.
  1. Extracting meaning (semantic information) of sentence.
  2. Using extracted information for detection.

- Meaning of sentence depend on:
  - Words Sequence.
    - Eg: [ram hit shyam] and [shyam hit ram]
  - Words meaning.
    - Eg: [Cutting nails is a good habit] and
    - [Nails use to hang things on wall]
  - Words interdependency.
- So extracting sentence semantic is difficult and required different approach for combining words.
- For combining words their vector form is required that represent their semantic information.

- There are word semantic extraction model like CBOW and Skip-gram (Mikolov et al. 2013), Glove (Pennington et al. 2014), etc. for extracting word semantic information in vector form.

Motivation

- Intelligent text processing.
- Text clustering and categorization.
- Enable a computer to process and understand the language used by human beings.

- Developing a model for extracting semantic information of sentence and use it for Paraphrasing Detection Testing.

- Developing a method for generating intermediate vector representations by using word vectors.

- Using generated intermediate vectors representations perform the paraphrasing detection test on MSRP dataset (Quirk et al. 2004).

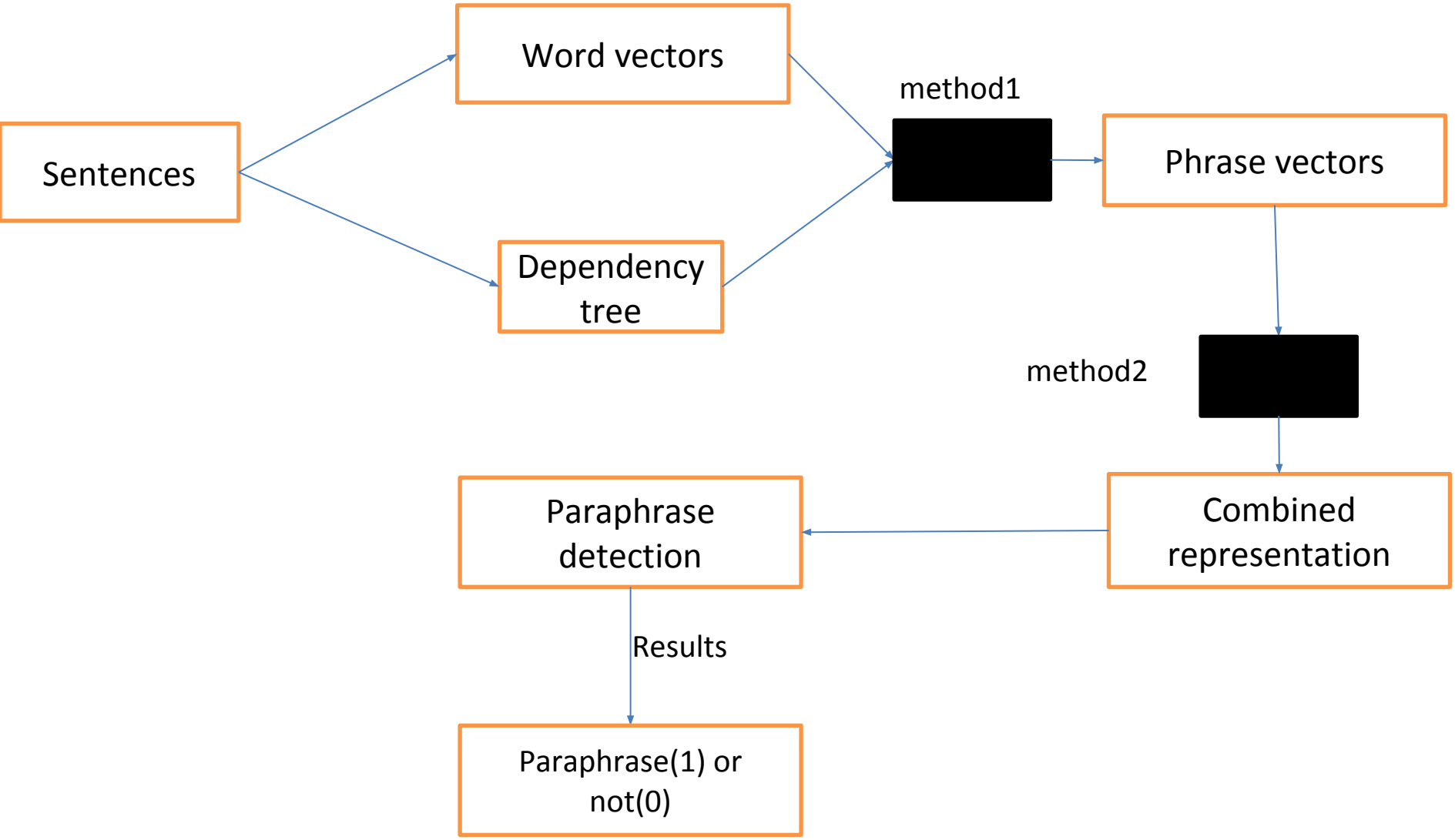
Recursive Autoencoder (RAE) 1990-2011

- **Pollack, J.B., 1990** Architecture to generate compact distributed representations for variable-sized tree-like structures.
- **Goller, C. and Kuchler, A., 1996** Backpropagation through structure for tree-like or recursive structures.
- **Socher, R., et al. 2011** RAE to learn vector representation of multi word phrase for sentiment prediction tasks.
- **Socher, R., et al. 2011** RAE and dynamic pooling for Paraphrasing detection.

Recursive Neural Network (RNN) 2010-2014

- **Socher, R., et al. 2010** RNN for parsing natural language and learning vector space representations for variable-sized inputs.
- **Socher, R., et al. 2011** RNN for parsing natural language sentence and images.
- **Socher, R., et al. 2012** RNN for learning compositional vector representations for phrases and sentences.
- **Socher, R., et al. 2014** Dependency Tree RNN model uses dependency tree to embed sentence into a vector space in order to retrieve images that are described by those sentences.

Rough process



word → word vector representation

Language model:

- Language model encapsulates the semantic information of the word in an appropriate representation.

1. **CBOW (Continuous Bag of Word) model:** (Mikolov et al. 2013)

- CBOW model is a one word context model, i.e. predict the word based on its context.

2. **Skip-gram model:** (Mikolov et al. 2013)

- Skip-gram model is opposite of CBOW model, i.e. takes one word as input and predicts it context word or surround the words.

word $\longrightarrow$ word vector representation

Word vector:

- Contain semantic information of word in the form of numerical vector.
- E.g.: word vector(xi) = $[0.112, 0.234, \ldots\ldots, -0.34]_n$

  where n length of vector

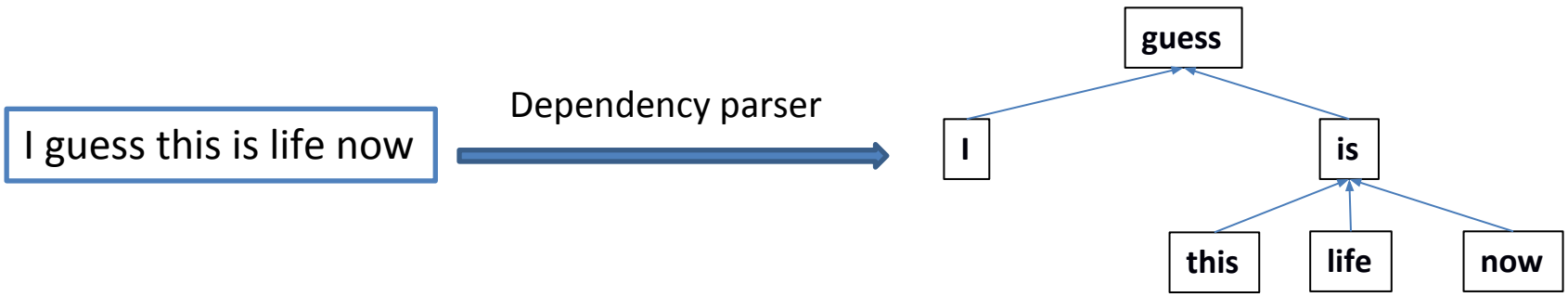
Sentence ⟶ Dependency tree

Dependency Parser: (Chen and Manning 2014)

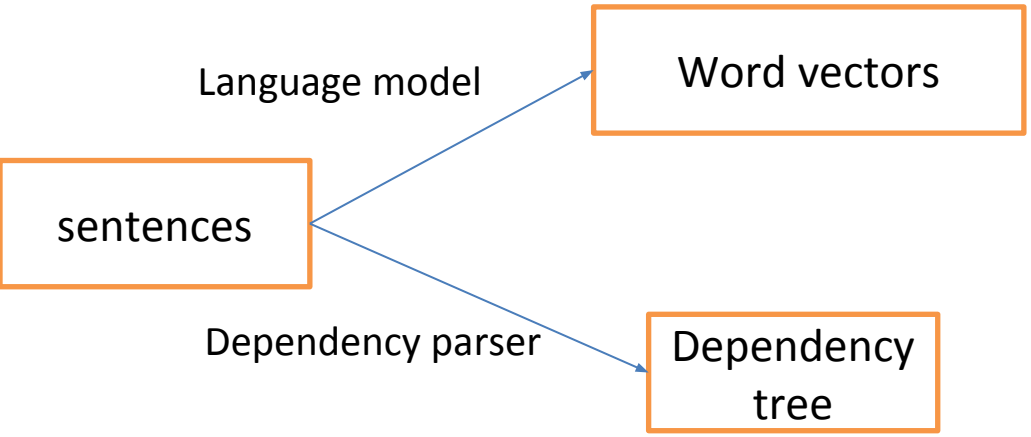
- Generate the dependency tree for the given sentence.

- A dependency tree maps a sentence to a tree in which each word is a node. Every node is either dependent on another node or the head of another node or both.

- E.g.:

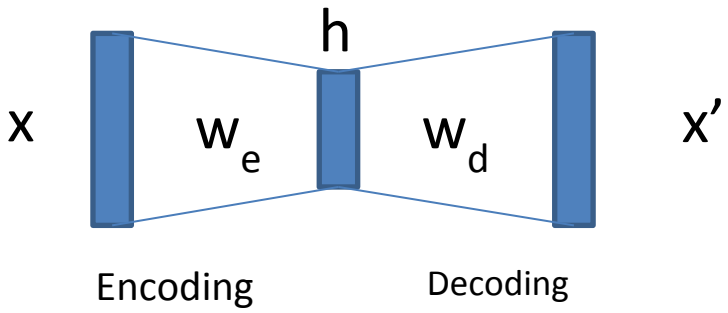


Partial Complete Process



Autoencoder

- Autoencoder (Hinton and Salakhutdinov 2006) is a special Neural network that uses input as its output for learning network weights.

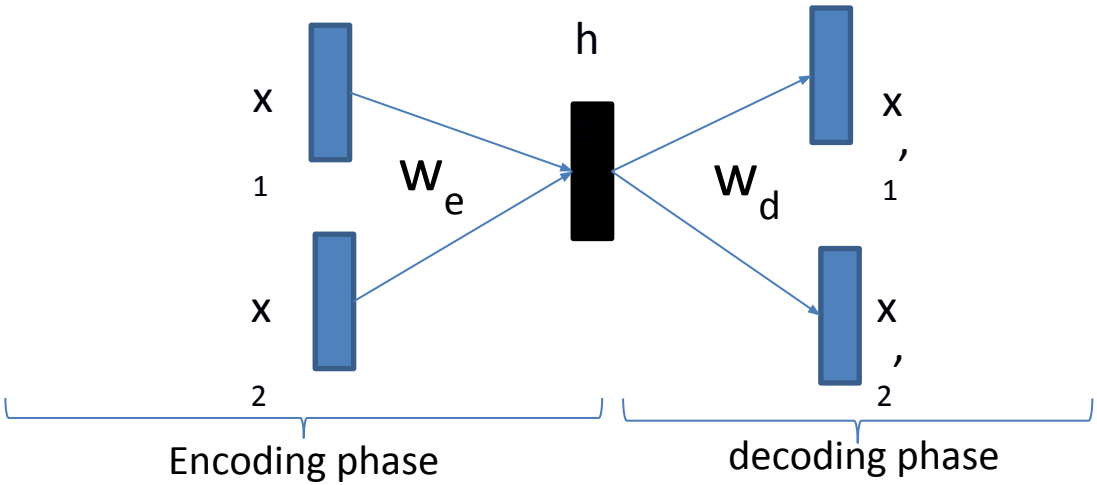


$$h = f(w_e.x), \quad x' = f(w_d.h), \quad \text{Error} = x - x'$$

- The main feature of an autoencoder is to encode given high dimension input into a low dimension form without loss of its important features.

Recursive Autoencoder

– Recursive Autoencoder (Socher et al. 2011c) is a tree structured autoencoder.



$h_1 = f(w_e.[x_1,x_2])$

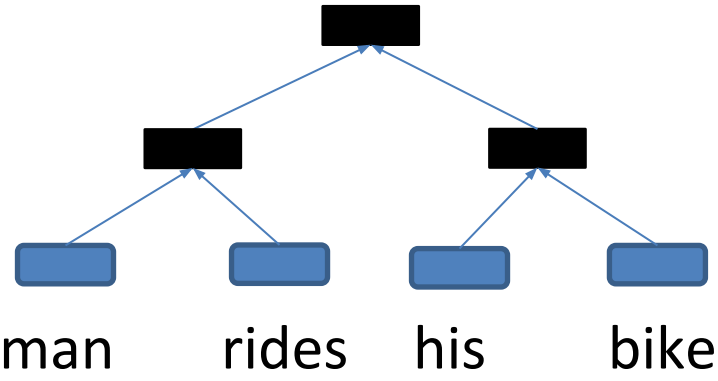
$[x_1', x_2'] = f(w_d.h_1)$

$Error = [x_1 - x_1'] + [x_2 - x_2']$

Recursive Autoencoder

- RAE only takes binary recursive form as input.

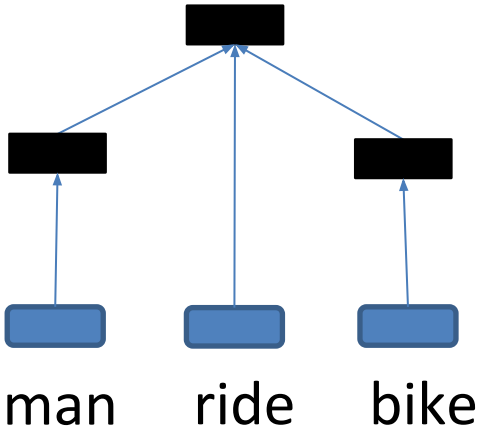
  e.g. for phrase [man rides his bike]



- Due to its binary recursive form acceptance its not suitable for recursive structure containing more then two child.

Dependency-Tree Recursive Neural Network

- DT-RNN (Socher, R. et al. 2014) to perform learning for tree-like structure.
- Can take recursive structure contain more then two child.
- E.g.: DT-RNN structure for phrase [man ride bike]



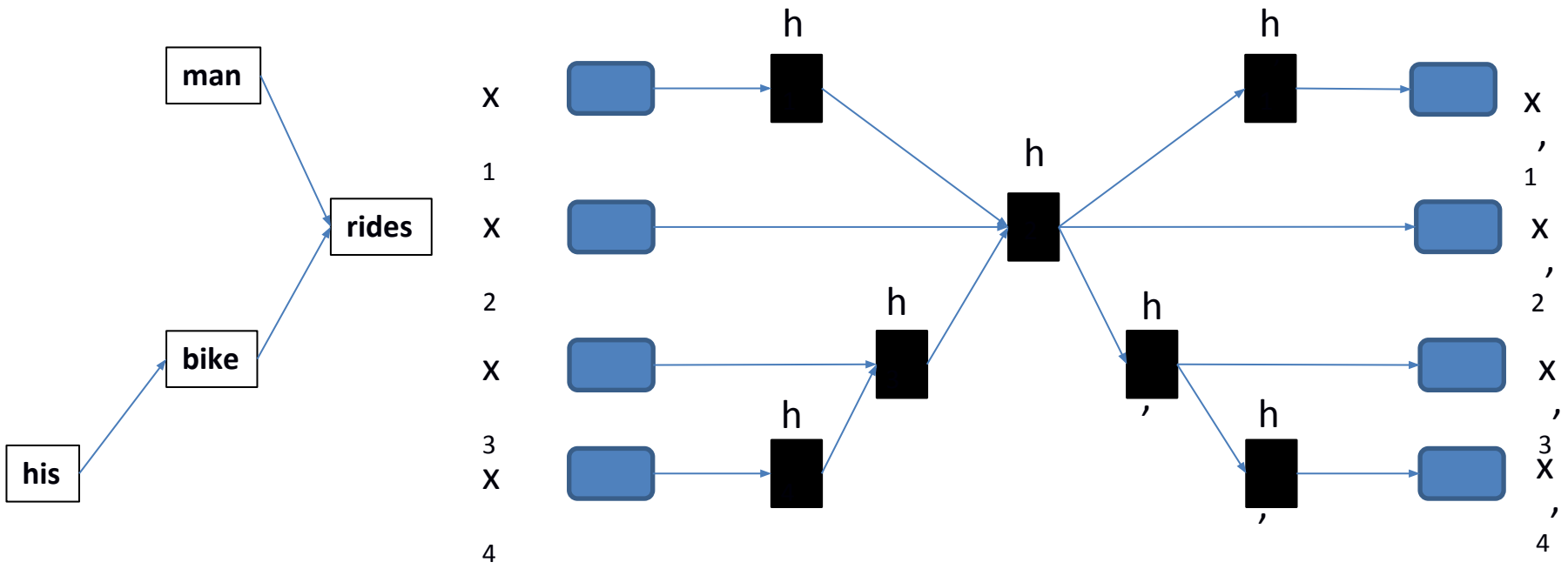
Dependency tree recursive autoencoder(DT-RAE)

- DT-RAE is an autoencoder formed by adding DT-RNN in Recursive autoencoder.
- DT-RAE is a recursive autoencoder for intermediate/phrase vectors representation generation.
- Uses dependency tree for generating recursive structure of sentence.
- Hidden or intermediate units are intermediate/phrase vectors representation.
- Decoding weights are taken as transpose of encoding weights.

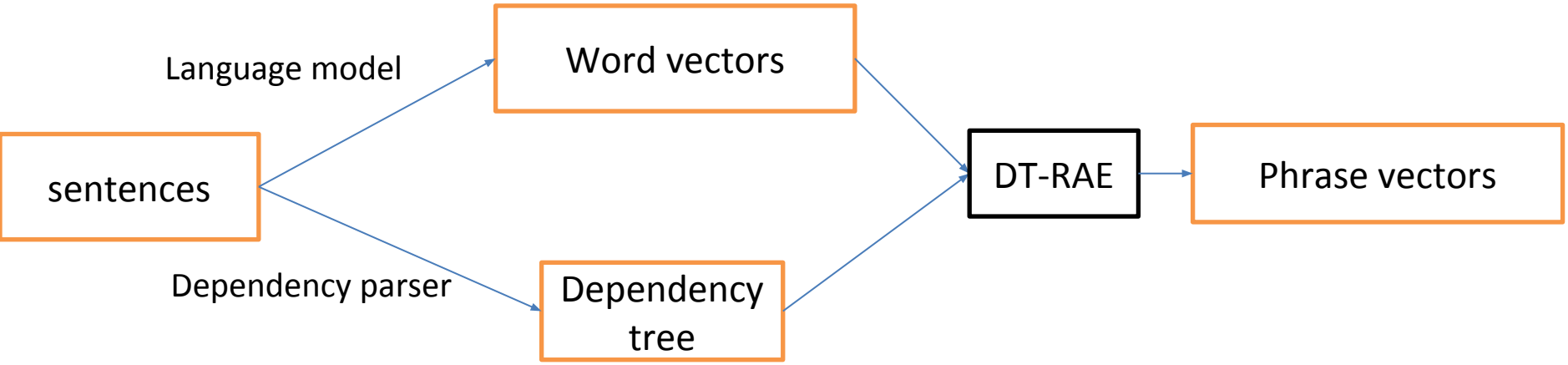
DT-RAE structure

Dependency tree

Recursive autoencoder



Partial Complete Process



Phrase vectors → Combined representation

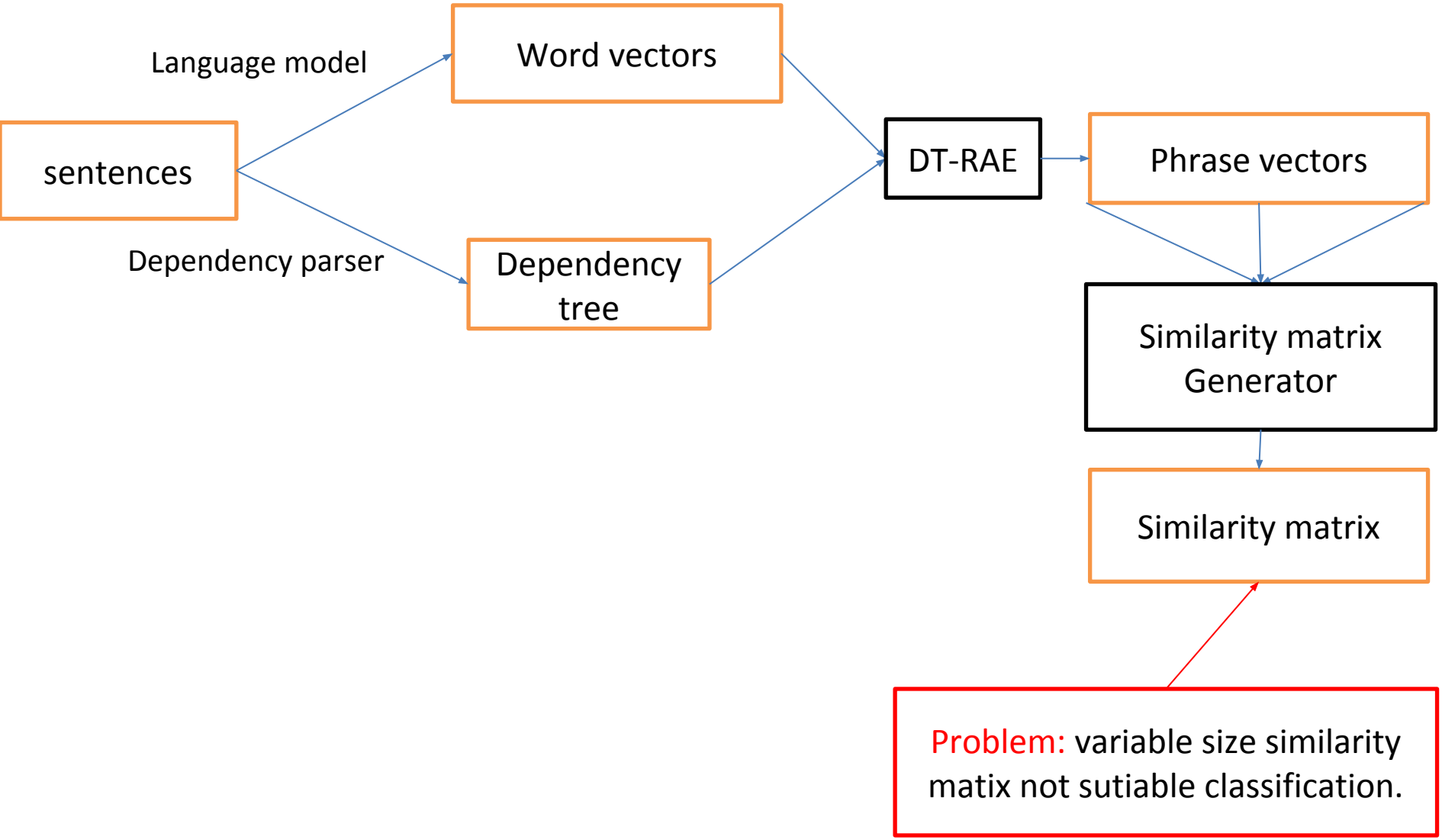
Similarity matrix:

- It takes two list of vector and return a matrix contain their similarity value.
  - Similarity matrix = $[e_{ij}]_{nxm}$
  - Where e_{ij} is Euclidian distance between v_i and u_j vector

- Similarity value = Euclidian distance, cosine value, etc.

- It takes phrase vectors and return similarity matrix.

Partial Complete Process

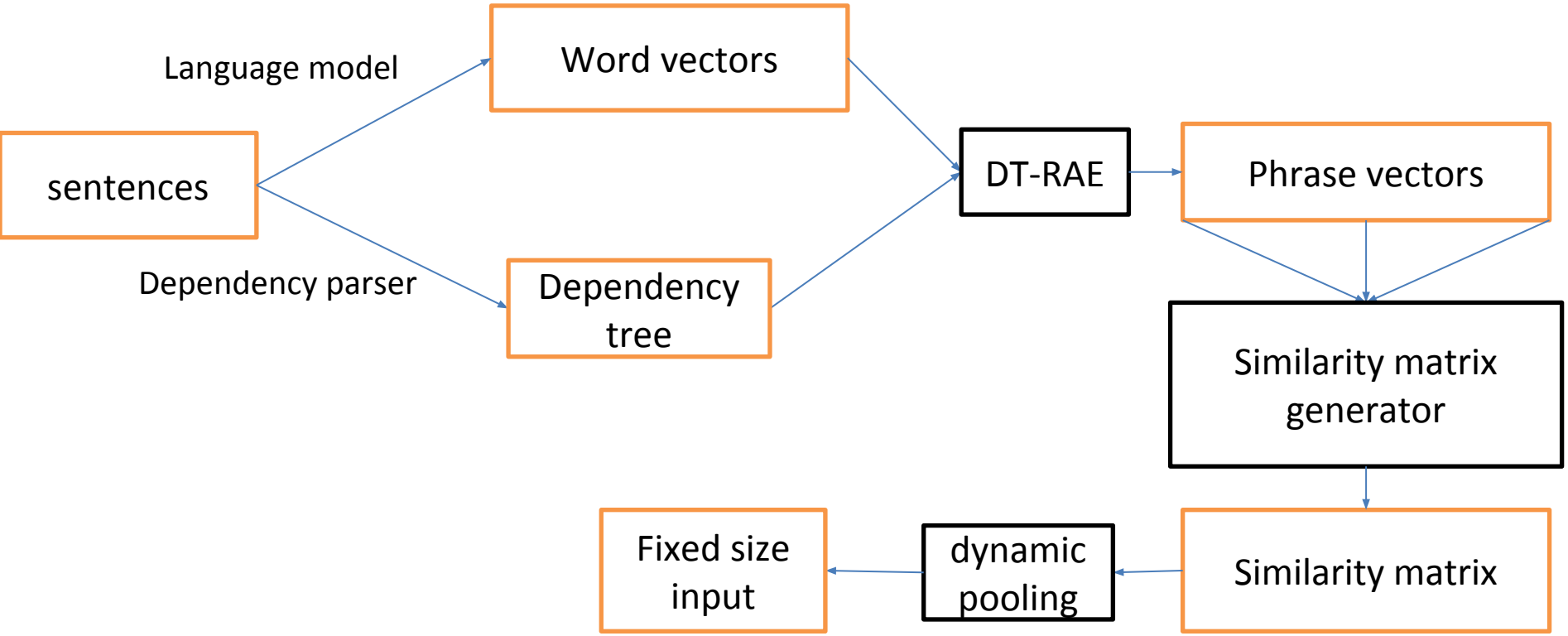


Variable size input ⟶ Fixed size input

Dynamic pooling: (Socher et al. 2011a)

- Convert a variable sized matrix (n*m) to fixed size matrix (p*p).
- Uses a pool function like min, max and mean.
- Perform replicating and pooling on sub-matrix to increase and decrease matrix size.

Complete Process



Extra feature

Number feature: (Socher et al. 2011a)

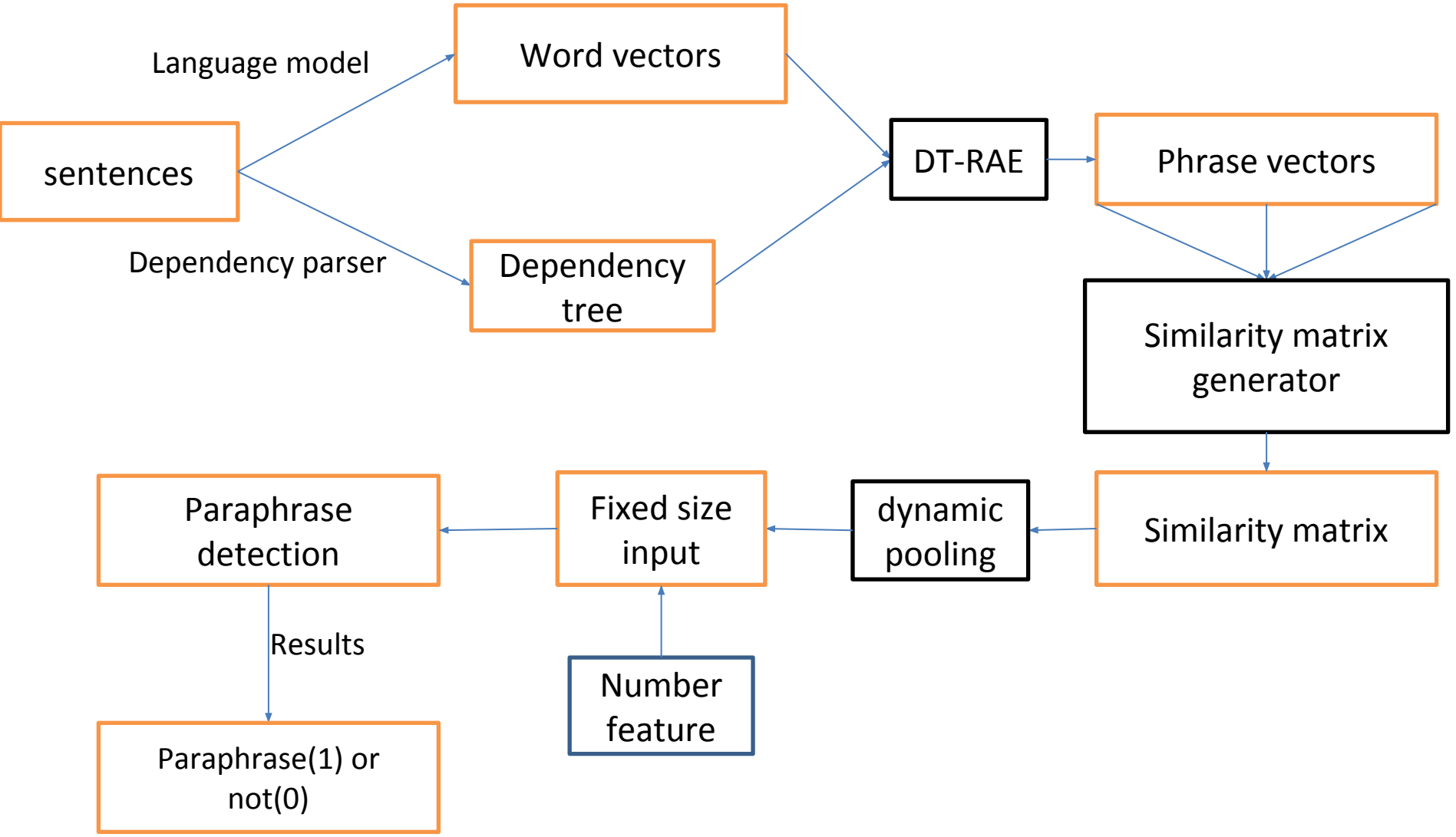
- Adds three values to the fixed input.
- First = 1, if the pair of sentences contains exactly same numbers or no number else 0.
- Second = 1, if both sentences contain the same numbers else 0.
- Third = 1, if the set of numbers in a sentence is a strict subset of the numbers in the other sentence, e.g. {12.3, 12, 20} and {12.3}.

Extra feature

Stopword Feature:

- Stopword are frequently occurring words and have little semantic value.
- Example is, a, was, etc.
- Stopword feature = 1 means stopwords include.
- Stopword feature = 0 means stopwords not include.

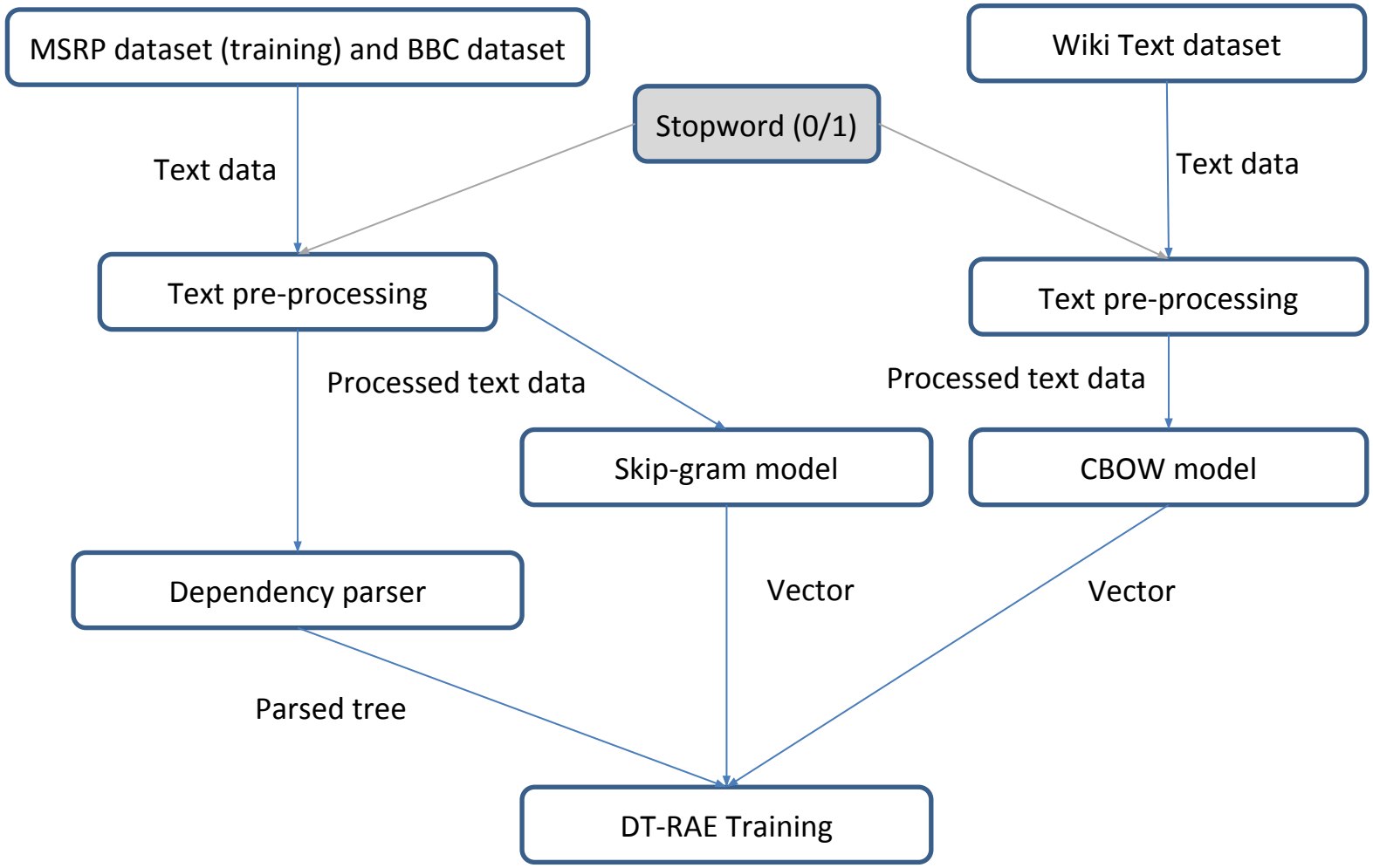
Complete Process



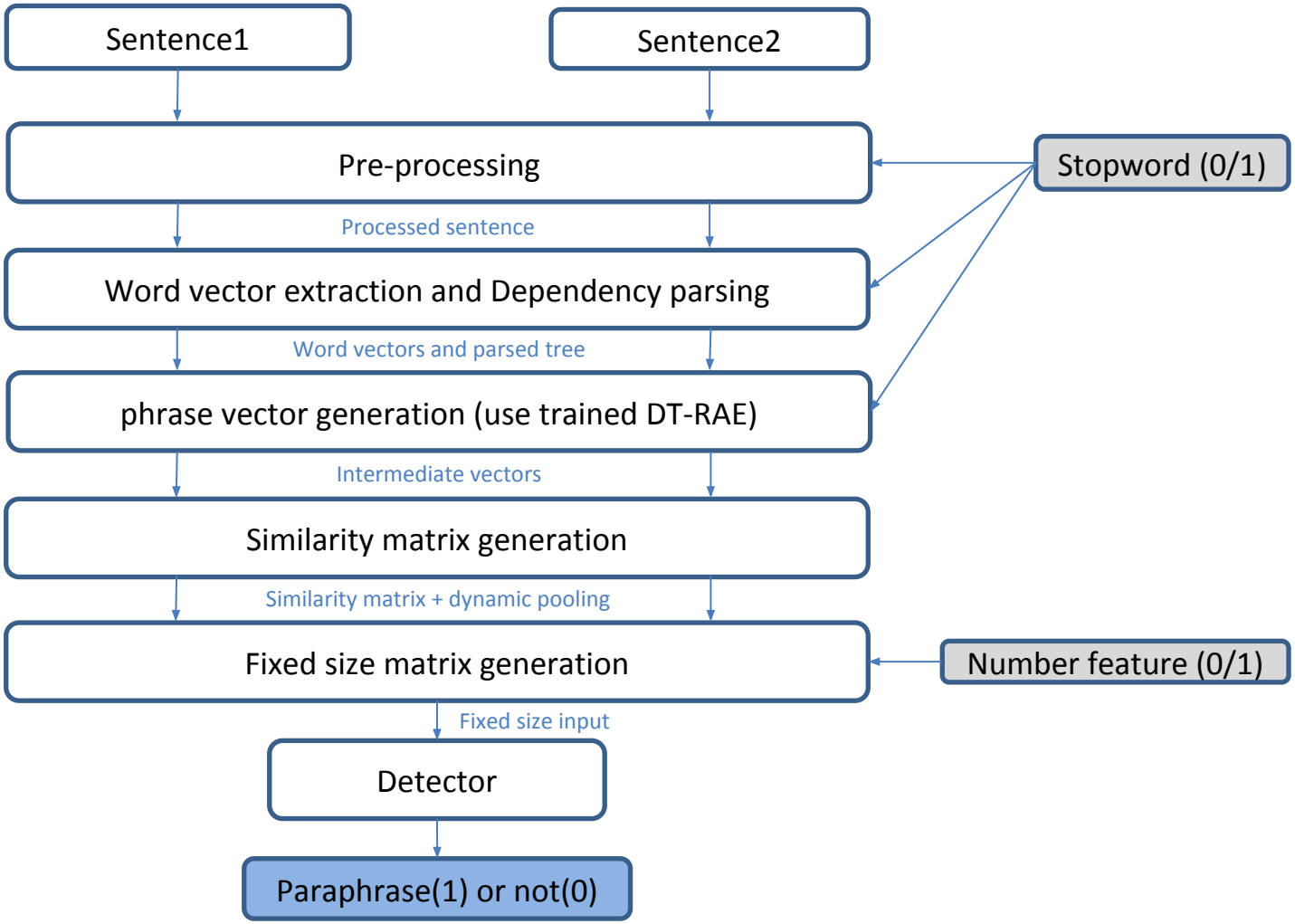
Dataset Description

Dataset	Source	size	Description	Reference
English Wikipedia dump	Wikipedia	8 million sentences	Wikipedia dumps are all available English text content in Wikipedia in an offline form.	English Wikipedia, 2017
BBC dataset	BBC News	37000 sentences	News article datasets, originating from BBC News.	D. Greene and P. Cunningham 2006
MSRP Corpus	Microsoft Corporation	5801 pairs sentences	Text file containing pairs of sentences extracted from news sources, along with human annotations indicating whether each pair is paraphrase or not	Quirk, C., C. Brockett, and W. B. Dolan. 2004

Training Flow of Proposed work



Testing Flow of Proposed work



DT-RAE Model description

- DT-RAE is trained using Gradient descent and reconstruction error.
- Decoding weights = transpose(encoding weight)
- Use the fixed values of learning rate and regularization controller on L2 regularization in weight update.
- The word vector size is 200.

DT-RAE Model Parameter description

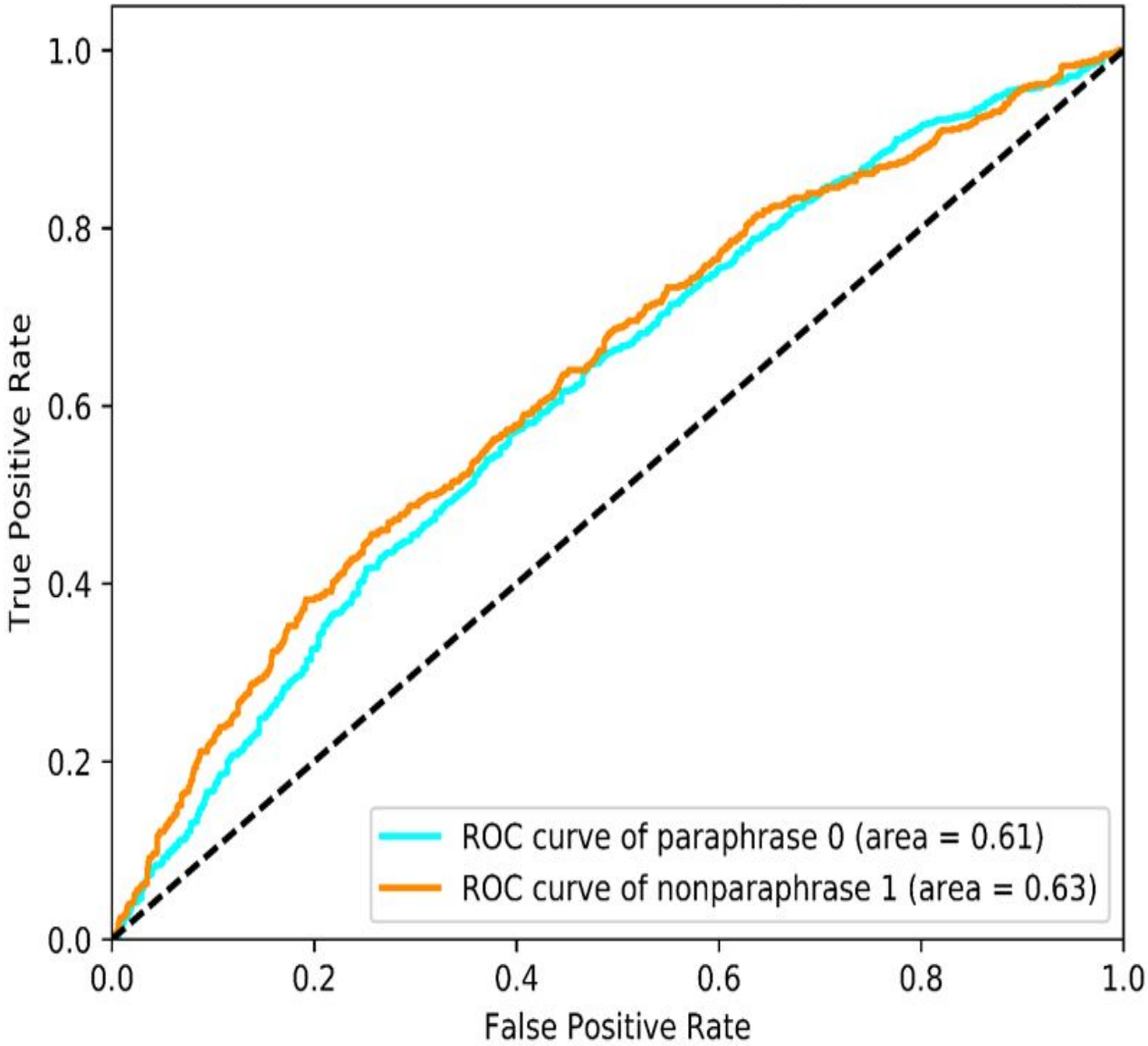
Model name	Stopword	No. of hidden lay.	Size of hidden lay.		Learning rate		Regularization Controller		Dynamic Pool size (For all experiment)
DT-RAE_h1_0	0	1	150		0.001		0.0001		10
DT-RAE_h1_1	1	1	150		0.001		0.0001		
DT-RAE_h2_0	0	2	Lay. 1	Lay. 2	Lay. 1	Lay. 2	Lay.1	Lay. 2	
			150	100	0.01	0.0005	0.05	0.0001	
DT-RAE_h2_1	1	2	Lay. 1	Lay. 2	Lay. 1	Lay. 2	Lay.1	Lay. 2	
			150	100	0.01	0.0005	0.05	0.0001	

Classifier Selection results

classifier	Min. acc.	Resp. F1 score	Max. acc.	Resp. F1 score	Avg. acc	Avg. F1 score
NN+log loss+lbfgs+tanh	60.058	69.561	61.275	71.4285	60.533	70.786
NN+log loss+adam+tanh	65.681	74.171	69.101	79.954	68.278	78.599
NN+log loss+adam+sigmoid	67.652	79.709	68.869	79.667	68.371	79.594
SVM + log loss + rbf	66.493	79.875	66.493	79.875	66.493	79.875
SVM+log loss+sigmoid	66.029	79.395	66.029	79.395	66.029	79.395
SVM+log loss+linear	68.696	79.405	68.695	79.405	68.696	79.405

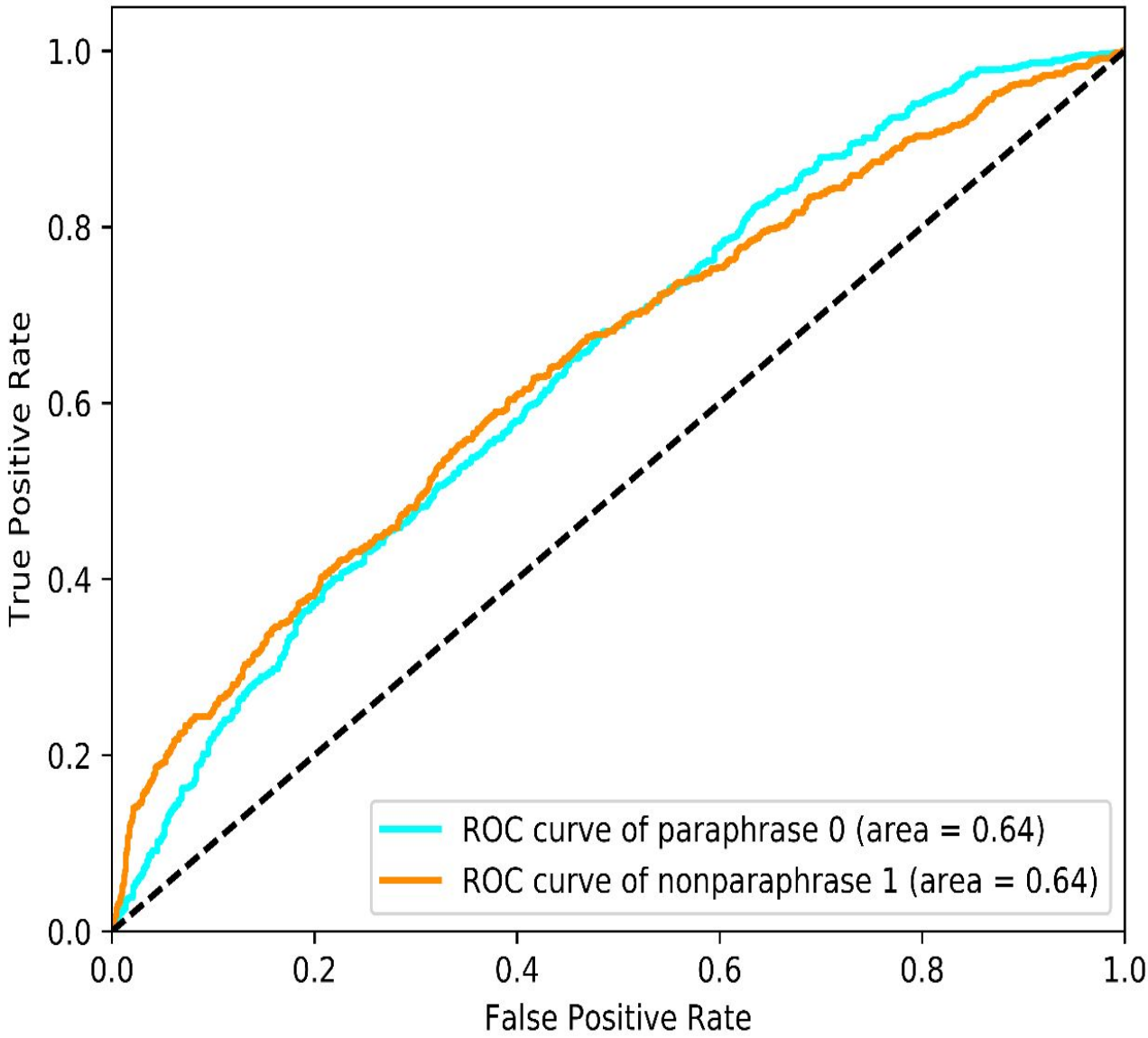
ROC curve Using

- **Model : DT-RAE_h1_0**
- **Stopword : 0**
- **Number feature : 0**



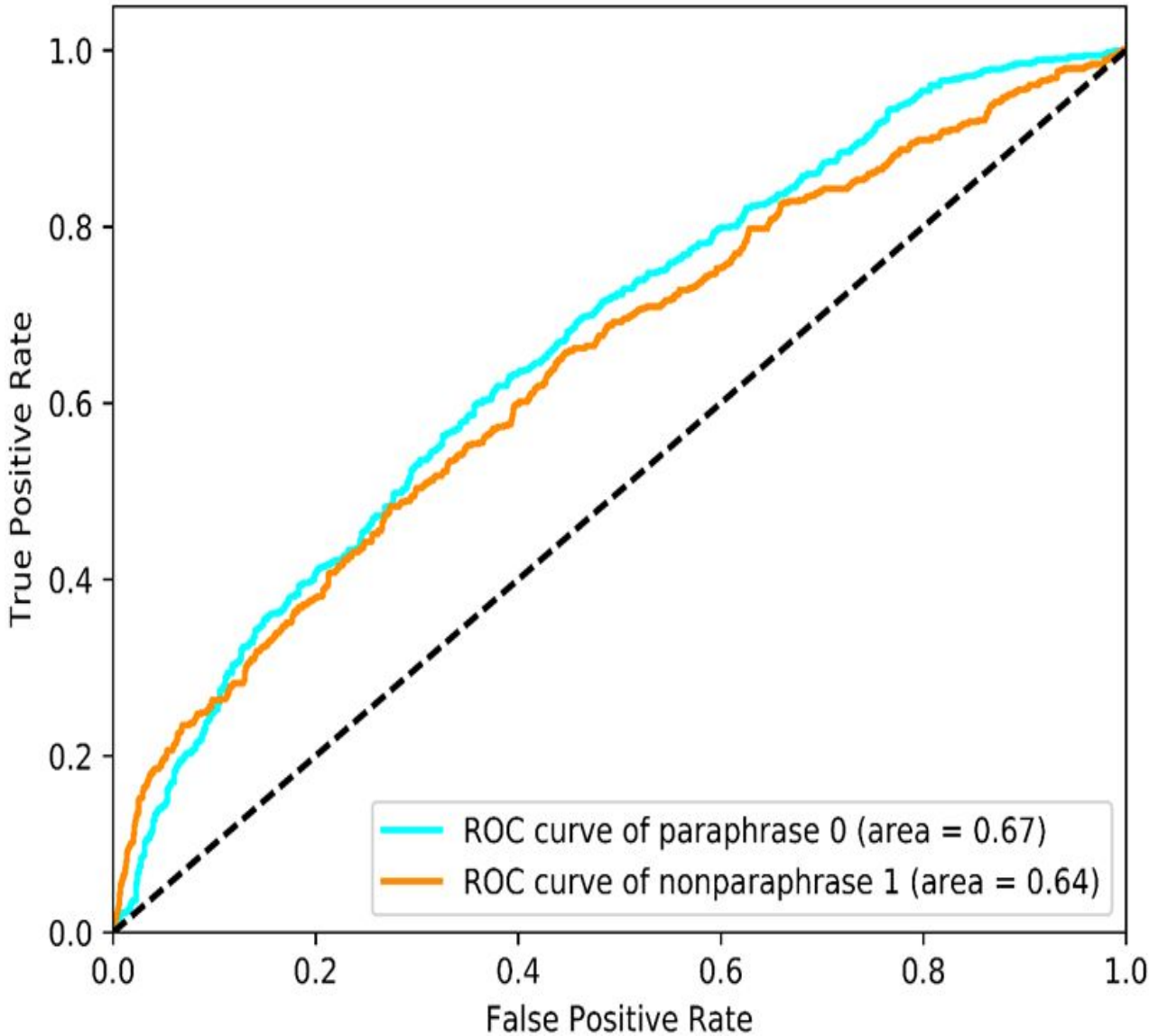
ROC curve Using

- **Model : DT-RAE_h1_1**
- **Stopword : 0**
- **Number feature : 1**



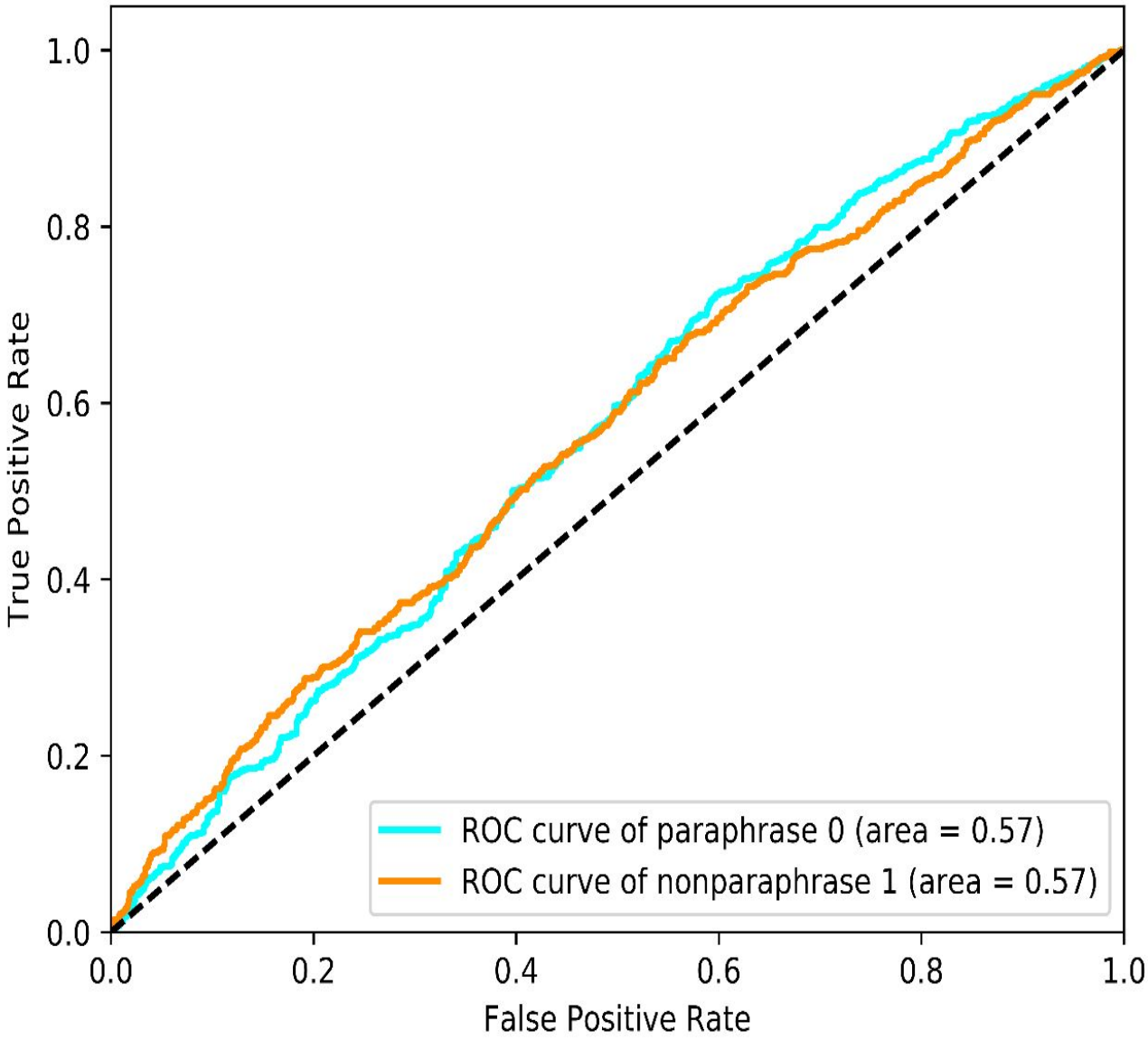
ROC curve Using

- **Model : DT-RAE_h1_1**
- **Stopword : 1**
- **Number feature : 1**



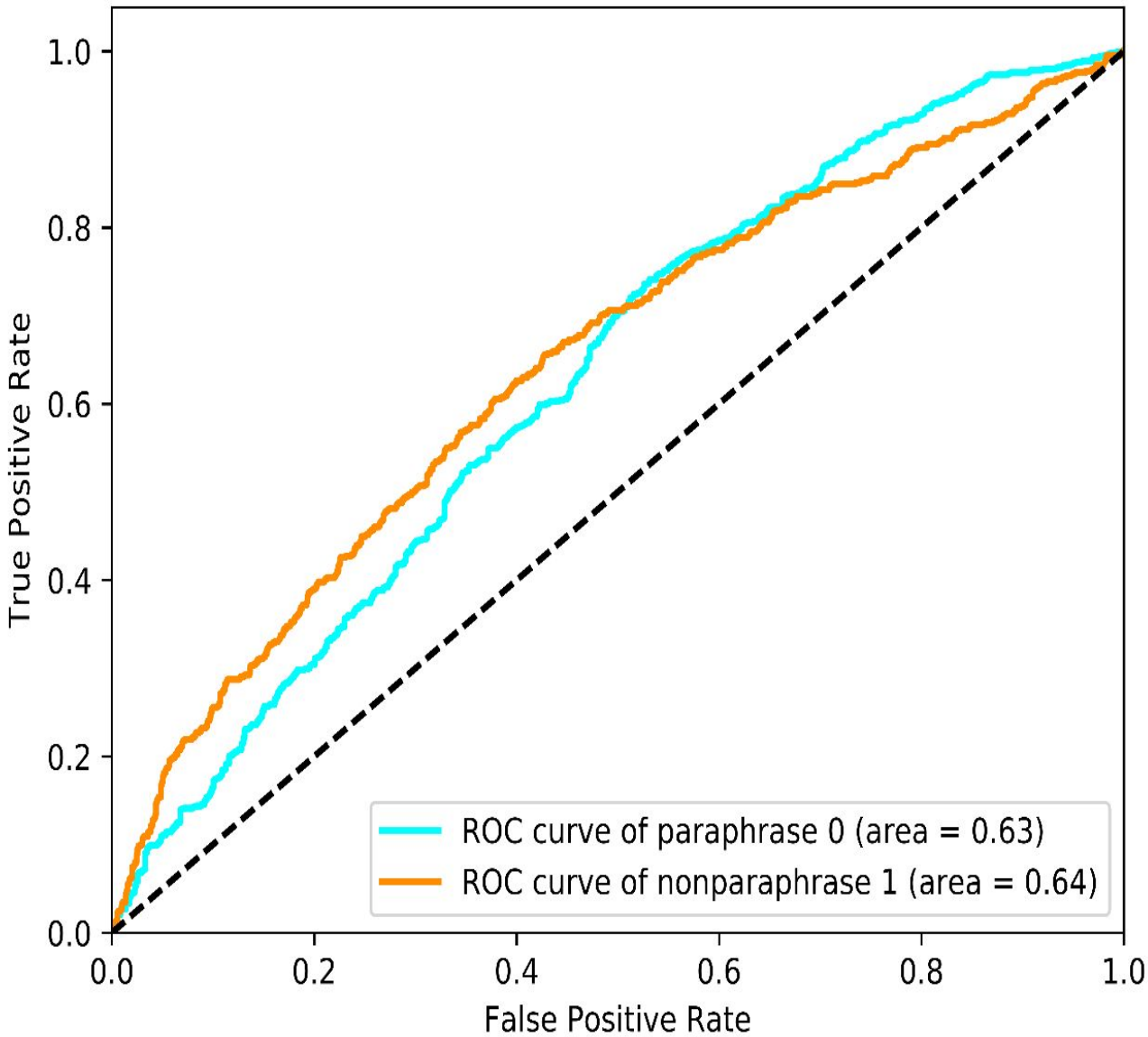
ROC curve Using

- **Model : DT-RAE_h2_0**
- **Stopword : 0**
- **Number feature : 0**



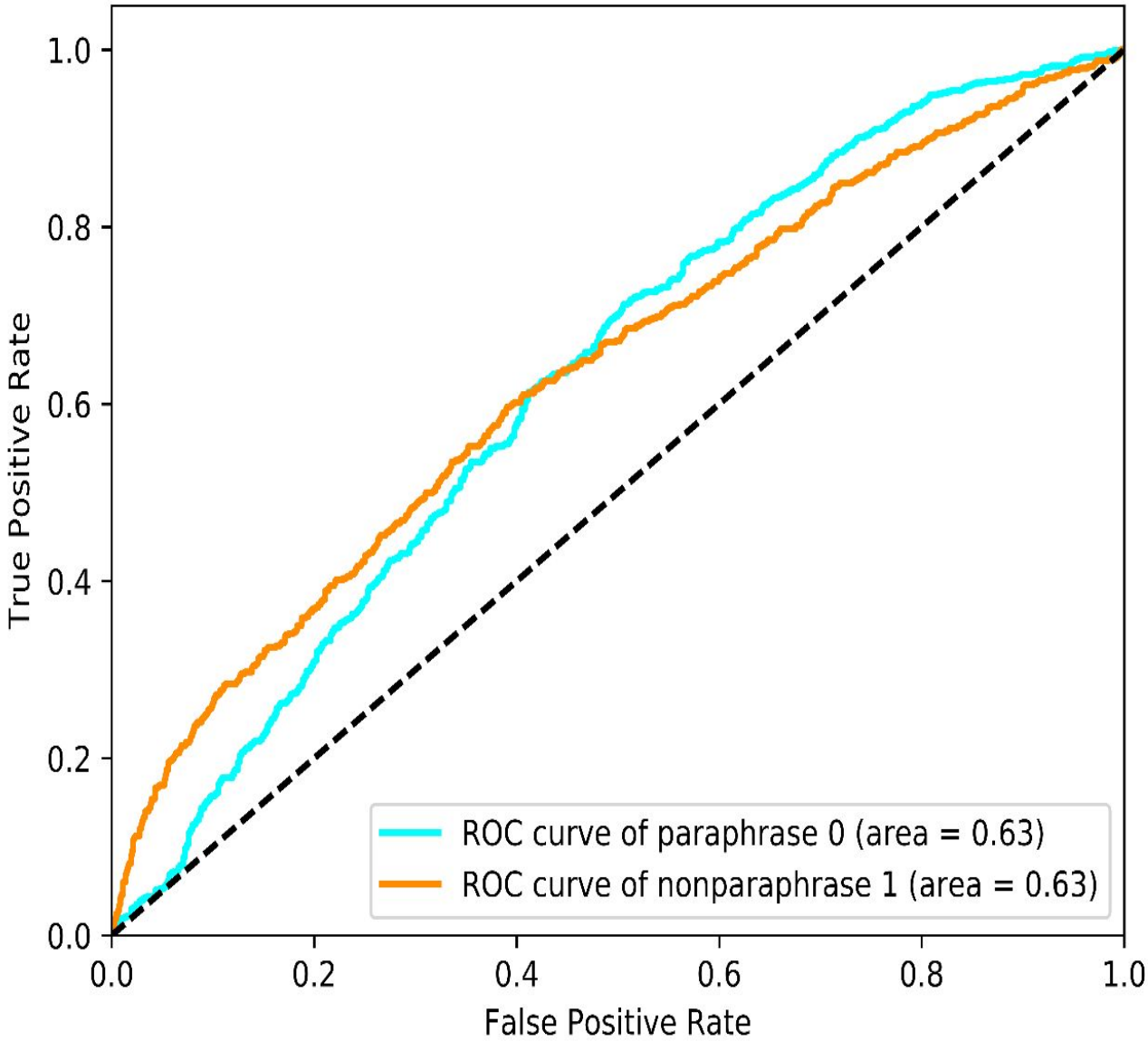
ROC curve Using

- **Model : DT-RAE_h2_1**
- **Stopword : 0**
- **Number feature : 1**



ROC curve Using

- **Model : DT-RAE_h2_1**
- **Stopword = 1**
- **Number feature = 1**



Results for various setting of proposed work

Model	Num. feature	Accuracy	F1 score
DT-RAE_h1_0 + dynamic pooling	0	66.493	79.875
DT-RAE_h1_0 + dynamic pooling	1	68.2899	79.082
DT-RAE_h1_1 + dynamic pooling	1	68.696	79.405
DT-RAE_h2_0 + dynamic pooling	0	66.493	79.875
DT-RAE_h2_0 + dynamic pooling	1	68.2899	79.082
DT-RAE_h2_1 + dynamic pooling	1	68.696	79.405

Comparison with other models for paraphrasing detection on MSRP dataset

Model	Reference	Acc.	F1 score
Vector Based Similarity (Baseline)	Mihalcea et al. (2006)	65.4	75.3
DT-RAE_h1_1 + dynamic pooling	Proposed model	68.69	79.4
RAE + dynamic pooling	Socher et al. (2011)	76.8	83.6

1. Training NN model for better generalization one should not aim for fast minimization of error.

2. There is still room for improvement by using advance optimization method.

Future Work

1. Application of Advance optimization method for weight updating can be applied .

2. Application of Deep learning in 2-layered DT-RAE model for better generalization can be applied.

3. Other language model for word vector can also be implemented.

Literature Cited

1. **Chen, D. and Manning, C.D., 2014.** A Fast and Accurate Dependency Parser using Neural Networks. In *EMNLP*, pp. 740-750.
2. **English Wikipedia, 2017.** Retrieved April 12, 2017, http://wikipedia.org/wiki/English_Wikipedia.
3. **Goller, C. and Kuchler, A., 1996.** Learning task-dependent distributed representations by backpropagation through structure. In Neural Networks, 1996., *IEEE International Conference* on (Vol. 1, pp. 347-352)
4. **Hinton, G.E. and Salakhutdinov, R.R., 2006.** Reducing the dimensionality of data with neural networks. *science*, *313*(5786), pp. 504-507.
5. **Mikolov, T., Chen, K., Corrado, G. and Dean, J., 2013**. Efficient estimation of word representations in vector space. *ArXiv preprint arXiv:1301.3781*.
6. **Pennington, J., Socher, R. and Manning, C.D., 2014.** Glove: Global Vectors for Word Representation. In *EMNLP*, Vol. 14, pp. 1532-1543.
7. **Pollack, J.B., 1990**. Recursive distributed representations. *Artificial Intelligence, 46(1)*, pp.77-105.

Literature Cited

9. **Socher, R., Manning, C.D. and Ng, A.Y., 2010**. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In Proceedings of the NIPS-2010 (pp. 1-9).
10. **Socher, R., Huang, E.H., Pennin, J., Manning, C.D. and Ng, A.Y., 2011a**. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *In Advances in Neural Information Processing Systems* (pp. 801-809).
11. **Socher, R., Lin, C.C., Manning, C. and Ng, A.Y., 2011b**. Parsing natural scenes and natural language with recursive neural networks. *In Proceedings of the 28th international conference on machine learning (ICML-11)* (pp. 129-136).
12. **Socher, R., Pennington, J., Huang, E.H., Ng, A.Y. and Manning, C.D., 2011c**. Semi-supervised recursive autoencoders for predicting sentiment distributions. *In Proceedings of the conference on empirical methods in natural language processing* (pp. 151-161). Association for Computational Linguistics.

Literature Cited

13. **Socher, R., Huval, B., Manning, C.D. and Ng, A.Y., 2012**. Semantic compositionality through recursive matrix-vector spaces. *In Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning* (pp. 1201-1211). Association for Computational Linguistics.
14. **Socher, R., Karpathy, A., Le, Q.V., Manning, C.D. and Ng, A.Y., 2014**. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics, 2*, pp.207-218.

Thank You