CS779 Project

Adversarial Techniques In NLP

Dept. of CSE, IIT Kanpur

Under the guidance of

Dr. Ashutosh Modi
Assistant Professor
Dept. of CSE

Group 3:

Abhishek Krishna- 20111002
Deeksha Arora- 20111017
Preeti Singh- 20111044
Sambrant Maurya- 20111054
Shruti Sharma- 20111061

Adversarial attacks on text

Original Input	Connoisseurs of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Positive (77%)**
Adversarial example [Visually similar]	**Aonnoisseurs** of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Negative (52%)**
Adversarial example [Semantically similar]	Connoisseurs of Chinese **footage** will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Negative (54%)**

Image: https://towardsdatascience.com/what-are-adversarial-examples-in-nlp-f928c574478e

Adversarial attacks on text

Original Input	Connoisseurs of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Positive (77%)**
Adversarial example [Visually similar]	**Aonnoisseurs** of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Negative (52%)**
Adversarial example [Semantically similar]	Connoisseurs of Chinese **footage** will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Negative (54%)**

- Neural Networks are susceptible to adversarial inputs.

Textual adversaries: Try to "fool" the Neural Network by introducing perturbations in the input text.

Image: https://towardsdatascience.com/what-are-adversarial-examples-in-nlp-f928c574478e

Motivation

Real Comment: admitting im not going to read this (...)	
Adv-comment: *hes a conservative from a few months ago*	
Prediction Change: Real News $\longrightarrow$ Fake News	

Image 1: An adversarial comment causes misclassification of a fake news detector

Motivation


NEWS

Man arrested after 'good morning' post mistranslated by Facebook as 'attack them'

Israeli police arrested a Palestinian man after his "good morning" post was translated by Facebook as "attack them."

Image 1: An adversarial comment causes misclassification of a fake news detector

Image 2: Facebook NMT mistakes input word for another which differs by a single character in Arabic, and wrecks havoc

Image 1: https://arxiv.org/pdf/2009.01048.pdf, Image 2: Article from theguardian.com

Motivation

Real Comment: admitting im not going to read this (...)	
Adv-comment: *hes a conservative from a few months ago*	
Prediction Change: Real News ⟶ Fake News	

NEWS

Man arrested after 'good morning' post mistranslated by Facebook as 'attack them'

Israeli police arrested a Palestinian man after his "good morning" post was translated by Facebook as "attack them."

Image 1: An adversarial comment causes misclassification of a fake news detector

Image 2: Facebook NMT mistakes input word for another which differs by a single character in Arabic, and wrecks havoc

Defence of Neural Networks against adversarial attacks is crucial.

Image 1: https://arxiv.org/pdf/2009.01048.pdf, Image 2: Article from theguardian.com

Existing Research

- A recent and very active research area

Existing Research

- A recent and very active research area
- Some latest attack methods:
  - DeepWordBug, Gao et.al (2018)[1]
  - TextBugger, Li at.al. (2018)[2]
  - *Textfooler, Jin et.al. (2019)[3]
  - CAM-RWR (Pruthi et.al.,2019)[4]
  - *BERT-Attack, Li et.al (2020)[5]
  - Bae, Garg et.al (2020)[6]
  - *Adv-OLM, Malik et.al (2021)[7]

*Performance benchmarks

Designing Textual attacks

- A text attack is built on two main components:
  - **Search methods:** Finding which characters/words/sentences to perturb.
  - **Transformations:** Replacing the chosen characters/words/sentences with an aim to cause misprediction of the target model.

Designing Textual attacks

- A text attack is built on two main components:
  - **Search methods:** Finding which characters/words/sentences to perturb.
  - **Transformations:** Replacing the chosen characters/words/sentences with an aim to cause misprediction of the target model.
- Desirable properties of a good attack method:
  - Accuracy under attack: *low*
  - Attack success rate: *high*
  - Perturbation percentage: *low*
  - Avg no of queries: *low*

Designing Textual attacks

- A text attack is built on two main components:
  - **Search methods:** Finding which characters/words/sentences to perturb.
  - **Transformations:** Replacing the chosen characters/words/sentences with an aim to cause misprediction of the target model.
- Desirable properties of a good attack method:
  - Accuracy under attack: *low*
  - Attack success rate: *high*
  - Perturbation percentage: *low*
  - Avg no of queries: *low*
- Most of the existing attacks don't score high on all 4 properties
  - Task at hand: Design an attack that does!

Experiments

- Tried transformations such as Misspelling Oblivious Word (MOE), GloVe, Word2Vec, FastText and finally concluded to use FastText.

- Experimenting on search-methods available in TextAttack framework:

  - **Greedy Search**: Rapid, but poor performance

  - **Beam Search**: Descent performance, but slow for larger text inputs

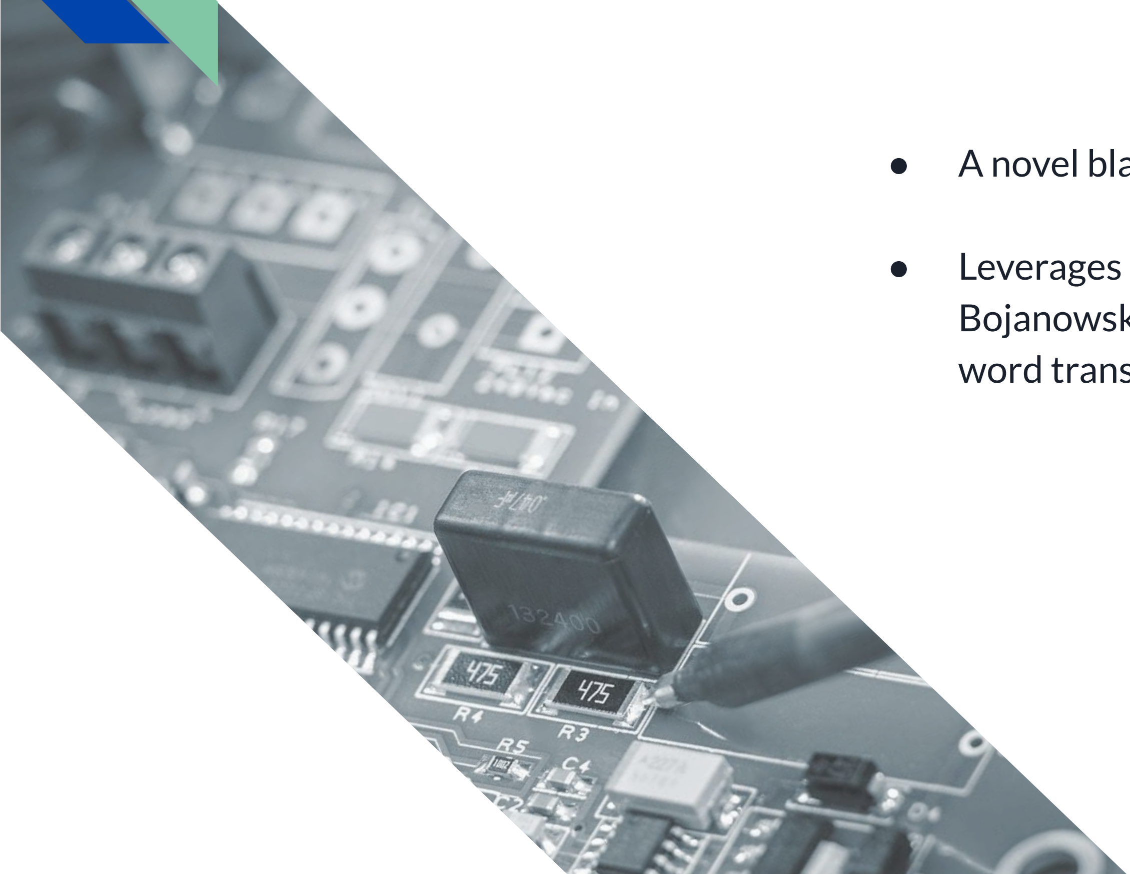
  - **Alzantot genetic algorithm[12]**: Poor performance, slow

  - **Particle Swarm Optimization[11]**: Best performance, but the slowest

  - **GreedyWordSwapWIR[9]**: Good performance and fast
    - Provides a good tradeoff between performance and speed

- Compared our attack's performance against existing recipes.

Introducing “FastAttack”



- A novel blackbox attack
- Leverages “FastText” embeddings by Bojanowski et.al[8], (Facebook Research) for word transformations

FastText Embeddings

- FastText: Open source, free, light-weighted library, developed by Facebook Research.
- Allows to learn text representation and text classifiers.

FastText Embeddings

- FastText: Open source, free, light-weighted library, developed by Facebook Research
- Allows to learn text representation and text classifiers.

- Additional ability to obtain word vectors for out-of-vocabulary words.
- Two models for computing word representations*:
    - CBOW(Continuous Bag of Words)
    - Skip-gram

*https://fasttext.cc/docs/en/crawl-vectors.html

FastText Embeddings

- FastText: Open source, free, light-weighted library, developed by Facebook Research
- Allows to learn text representation and text classifiers.

- Additional ability to obtain word vectors for out-of-vocabulary words.
- Two models for computing word representations:
  - CBOW(Continuous Bag of Words)
  - Skip-gram

- CBOW: predict target word according to its context.
  - Context represented as BOW containing fixed size window around target.

https://fasttext.cc/docs/en/unsupervised-tutorial.html

FastText Embeddings

- FastText: Open source, free, light-weighted library, developed by Facebook Research
- Allows to learn text representation and text classifiers.

- Additional ability to obtain word vectors for out-of-vocabulary words.
- Two models for computing word representations:
    - CBOW(Continuous Bag of Words)
    - Skip-gram

- CBOW: predict target word according to its context.
    - Context represented as BOW containing fixed size window around target.

- Skip-gram: learns to predict target word thanks to a near-by word.
    - In practice, skip-gram models works better with subword information than cbow.

https://fasttext.cc/docs/en/unsupervised-tutorial.html

Embeddings used

- FastText, Word2Vec and GLoVE
    - Trained on the ***text8*** corpus*, consisting of first 100,000,000 bytes of plain text from Wikipedia

text8 corpus: http://mattmahoney.net/dc/textdata.html

Embeddings used

- FastText, Word2Vec and GLoVE
    - Trained on the ***text8*** corpus*, consisting of first 100,000,000 bytes of plain text from Wikipedia
- **Hyperparameters:**

Embedding	Model type	Vector size	Window size	Learning rate	Epochs	Threads	Training time (s)	Total words
FastText	Skipgram	100	5	0.025	5	3	932	17005207
Word2Vec	Skipgram	100	5	0.025	5	3	670	17005207
Glove	-	100	5	0.05	5	3	628	17005207

text8 corpus: http://mattmahoney.net/dc/textdata.html

Target models and Datasets

- **HuggingFace* models with datasets:**
  - **Albert-Base-V2**
    - AG News, dataset ag_news, split test
    - IMDB, dataset imdb, split test
    - Movie Reviews, dataset rotten_tomatoes, split test
    - Yelp Polarity, dataset yelp_polarity, split test
  - **Bert-Base-uncased**
    - AG News, dataset ag_news, split test
    - IMDB, dataset imdb, split test
    - Movie Reviews, dataset rotten_tomatoes, split test
    - Yelp Polarity, dataset yelp_polarity, split test

*https://huggingface.co/textattack

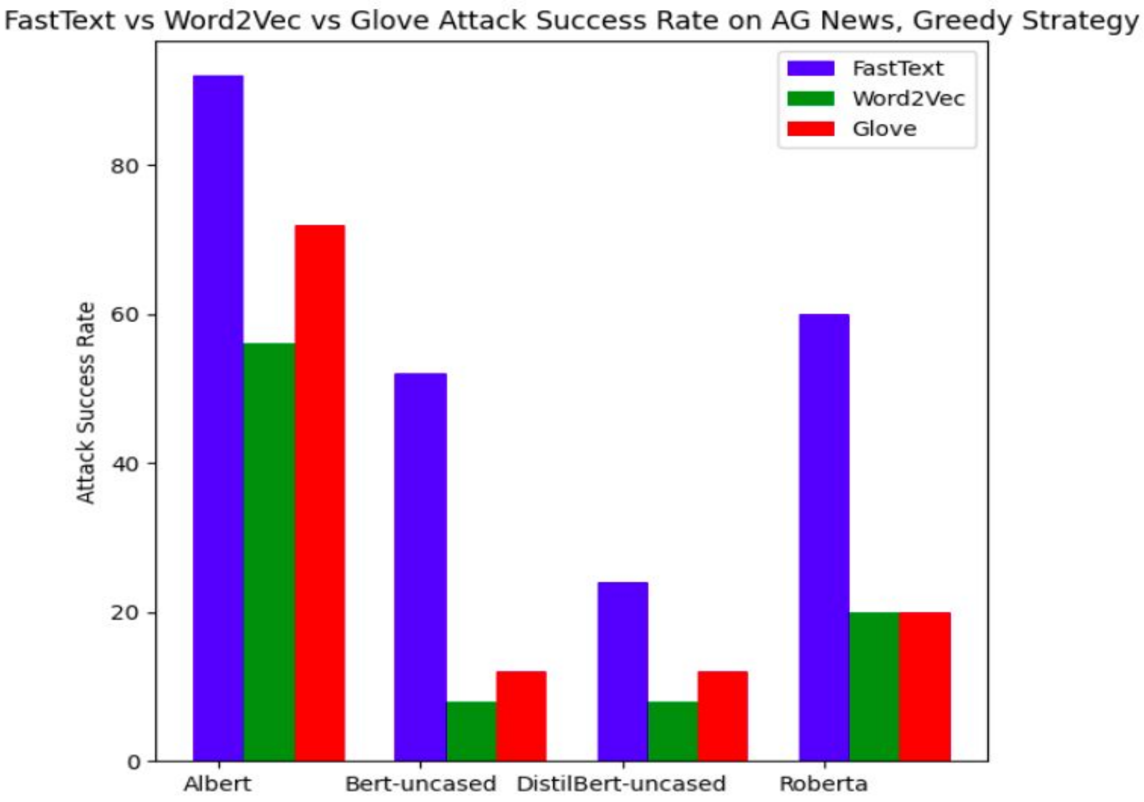
Target models and Datasets

- **HuggingFace* models with datasets:**

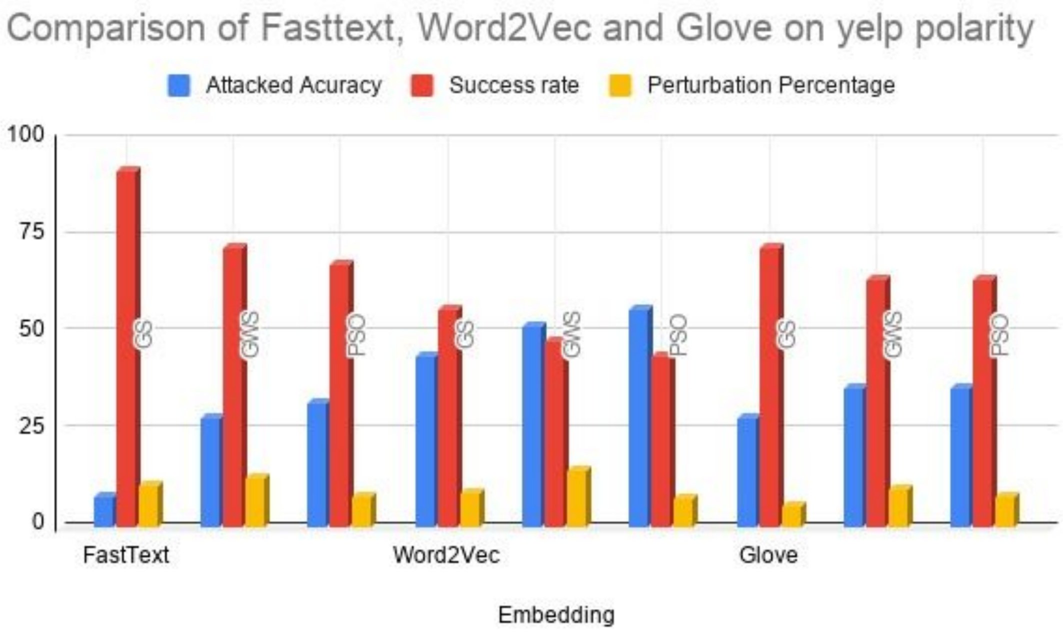
  - **Distilbert-Base-cased**
    - Quora Question Pairs, dataset glue, qqp, split validation
    - SST-2, dataset glue, sst2, split validation
  - **Distilbert-Base-uncased**
    - AG News, dataset ag_news, split test
    - IMDB, dataset imdb, split test
  - **Roberta-Base**
    - AG News, dataset ag_news, split test
    - IMDB, dataset imdb, split test
    - Movie Reviews, dataset rotten_tomatoes, split test
    - SST-2, dataset glue, sst2, split validation

*https://huggingface.co/textattack

FastText vs Word2Vec vs GloVE



FastText vs Word2Vec vs GloVE



Search methods: GS: Greedy Search, GWS: GreedyWordSwapWIR[9], PSO: ParticleSwarmOptimization[11]

FastAttack implementation

- **Search method:** Uses GreedyWordSwapWIR algorithm[9] with weighted saliency to find the words **W** to be perturbed.
    - A tradeoff between speed and accuracy

FastAttack implementation

- **Search method:** Uses GreedyWordSwapWIR algorithm[9] with weighted saliency to find the words **W** to be perturbed.

    - A tradeoff between speed and accuracy

- **Transformation technique:** Replace the chosen word **w** ∊ **W** with its closest neighbor **w'** in the FastText embedding under the following constraints:

    - **w'** should not have **w** as a substring.
    - **w** should not be a stop-word as defined in the NLTK corpora*.
    - For the same sentence, the same word should not be modified twice.
    - The case of the replaced word should be preserved.

*https://www.nltk.org/book/ch02.html

FastAttack implementation

- **Search method:** Uses GreedyWordSwapWIR algorithm[9] with weighted saliency to find the words **W** to be perturbed.

  - A tradeoff between speed and accuracy

- **Transformation technique:** Replace the chosen word **w** $\in$ **W** with its closest neighbor **w'** in the FastText embedding under the following constraints:

  - **w'** should not have **w** as a substring.
  - **w** should not be a stop-word as defined in the NLTK corpora.
  - For the same sentence, the same word should not be modified twice.
  - The case of the replaced word should be preserved.

- **Goal function:** UntargetedClassification

- We use the TextAttack framework, Morris et.al.[10] (2020) to frame our attack.

Performance of FastAttack

An Illustration

Sci/tech (98%) --> World (73%) **(FastAttack)**

E-mail scam targets police chief Wiltshire Police warns about "phishing" after its fraud squad chief was targeted.

E-mail scam targets police chief Wiltshire Police warns about "furnishing" after its bribery squad chief was targeted.

Sci/tech (98%) --> World (54%) **(TextFooler)**

E-mail scam targets police chief Wiltshire Police warns about "phishing" after its fraud squad chief was targeted.

E-mail scam targets police chief Wiltshire Police warns about "phishing" after its hoax battalion leiter was targeted.

An Illustration

Sci/tech (98%) --> World (73%) **(FastAttack)**

E-mail scam targets police chief Wiltshire Police warns about "phishing" after its fraud squad chief was targeted.

E-mail scam targets police chief Wiltshire Police warns about "furnishing" after its bribery squad chief was targeted.

Sci/tech (98%) --> World (54%) **(TextFooler)**

E-mail scam targets police chief Wiltshire Police warns about "phishing" after its fraud squad chief was targeted.

E-mail scam targets police chief Wiltshire Police warns about "phishing" after its hoax battalion leiter was targeted.

Sci/tech (98%) --> World (78%) **(DeepWordBug)**

E-mail scam targets police chief Wiltshire Police warns about "phishing" after its fraud squad chief was targeted.

E-mail scam targets police chief Wiltshire Police warns about "phishin" after its fruad squad chief was targeted.

Sci/tech (98%) --> World (82%) **(BERT-Attack)**

E-mail scam targets police chief Wiltshire Police warns about "phishing" after its fraud squad chief was targeted.

E-mail scam targets police chief Wiltshire Police warns about "phitism" after its fraudulent squad chief was targeted.

FastAttack vs CAM-RWR (Pruthi et.al.,2019)

Model	Dataset	Attack method	Successful attacks (Out of 25)	Original accuracy	Attacked accuracy	Success rate	Perturbed %	Avg num queries	Attack time (in seconds)
ALBERT-base-v2	YELP-polarity	fastattack	**19**	100%	**24%**	**76%**	9.47%	**241.72**	**139.27**
		cam-rwr	2		92%	8%	**4.28%**	1666.0	924.62
	IMDB	fastattack	**17**	89.29%	**28.57%**	**68.0%**	2.3%	**420.24**	**271.88**
		cam-rwr	8		60.71%	32.0%	**0.53%**	3843.64	2493.46
BERT-base-uncased	YELP-polarity	fastattack	**18**	100%	**28.0%**	**72.0%**	11.07%	**241.48**	**128.77**
		cam-rwr	1		96.0%	4.0%	**6.67%**	1665.96	848.89
	IMDB	fastattack	**20**	92.59%	**18.52%**	**80.0%**	5.61%	**428.4**	**251.87**
		cam-rwr	10		55.56%	40.0%	**0.48%**	3868.6	2192.64

FastAttack vs Faster Genetic Algorithm (Jia et.al.,2019)

Model	Dataset	Attack method	Successful attacks (Out of 25)	Original accuracy	Attacked accuracy	Success rate	Perturbed %	Avg num queries	Attack time (in seconds)
ALBERT-base-v2	YELP-polarity	fastattack	**19**	100%	**24%**	**76%**	**11.43%**	**242.76**	**114.78**
		FGA_Jia	17		32%	68%	11.6%	4762.84	5833.25
	IMDB	fastattack	**16**	96.15%	**34.62%**	**64.0%**	**17.56%**	**63.52**	**28.86**
		FGA_Jia	10		57.69%	40.0%	18.3%	3358.08	3900.52
BERT-base-uncased	YELP-polarity	fastattack	**21**	100%	**16%**	**84%**	12.51%	**238.84**	**100.20**
		FGA_Jia	18		28%	72%	**10.12%**	5123.12	4985.12
	IMDB	fastattack	**16**	92.59%	**33.33%**	**64.0%**	21.46%	**65.16**	**32.84**
		FGA_Jia	9		59.26%	36.0%	**15.85%**	3292.84	3728.22

FastAttack vs TextFooler (Jin et.al.,2019)

Model	Dataset	Attack method	Successful attacks (Out of 25)	Original accuracy	Attacked accuracy	Success rate	Perturbed %	Avg num queries	Attack time (in seconds)
ALBERT-base-v2	YELP-polarity	fastattack	19	100%	24.0%	76.0%	11.43%	**242.76**	**210.15**
		textfooler	**24**		**4.0%**	**96%**	**9.11%**	480.92	316.80
	IMDB	fastattack	22	89.29%	10.71%	88.0%	**5.2%**	**434.92**	**426.02**
		textfooler	**25**		0.0%	100%	8.35%	674.2	473.95
BERT-base-uncased	YELP-polarity	fastattack	21	100%	16.0%	84.0%	12.51%	**238.84**	**137.26**
		textfooler	**25**		**0.0%**	**100%**	**10.89%**	479.08	237.07
	IMDB	fastattack	21	92.59%	14.81%	84.0%	**5.29%**	**419.72**	**342.94**
		textfooler	**25**		**0.0%**	**100.0%**	7.99%	663.64	400.18

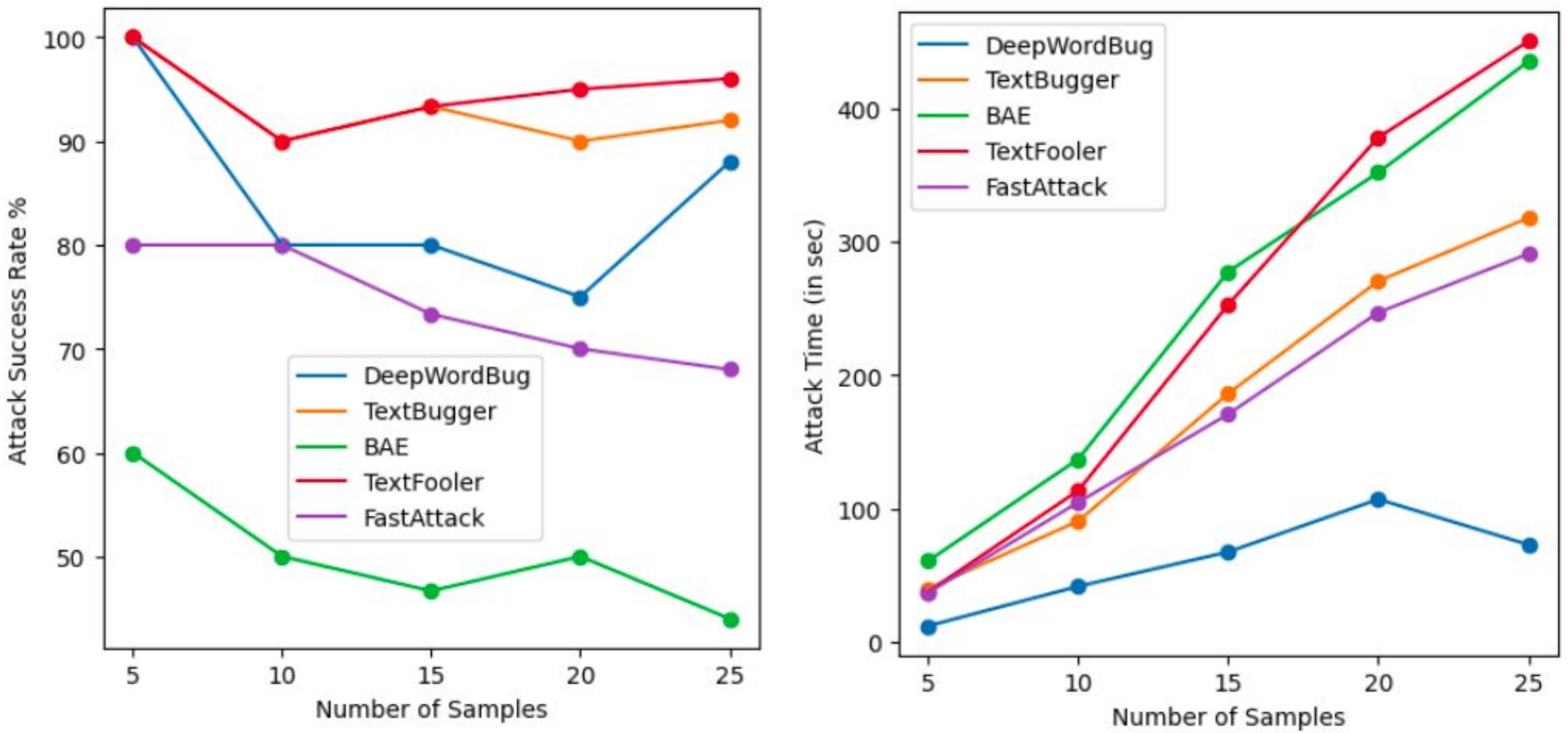
FastAttack vs BAE (Garg et.al.,2020)

Model	Dataset	Attack method	Successful attacks (Out of 25)	Original accuracy	Attacked accuracy	Success rate	Perturbed %	Avg num queries	Attack time (in seconds)
ALBERT-base-v2	YELP-polarity	fastattack	**17**	100%	**32%**	**68%**	8.19%	**242.28**	**146.80**
		BAE_garg	11		56%	44%	**4.72%**	287.24	438.28
	IMDB	fastattack	**22**	89.29%	**10.71%**	**88.0%**	5.2%	**434.92**	**584.61**
		BAE_garg	16		32.14%	64.0%	**2.73%**	512.32	1051.52
BERT-base-uncased	YELP-polarity	fastattack	**21**	100%	**16%**	**84%**	12.81%	**238.84**	**252.68**
		BAE_garg	11		56%	44%	**6.44%**	314.88	764.20
	IMDB	fastattack	**21**	92.59%	**14.81%**	**84.0%**	5.29%	**419.72**	**268.29**
		BAE_garg	12		48.15%	48.0%	**2.97%**	561.52	1153.58

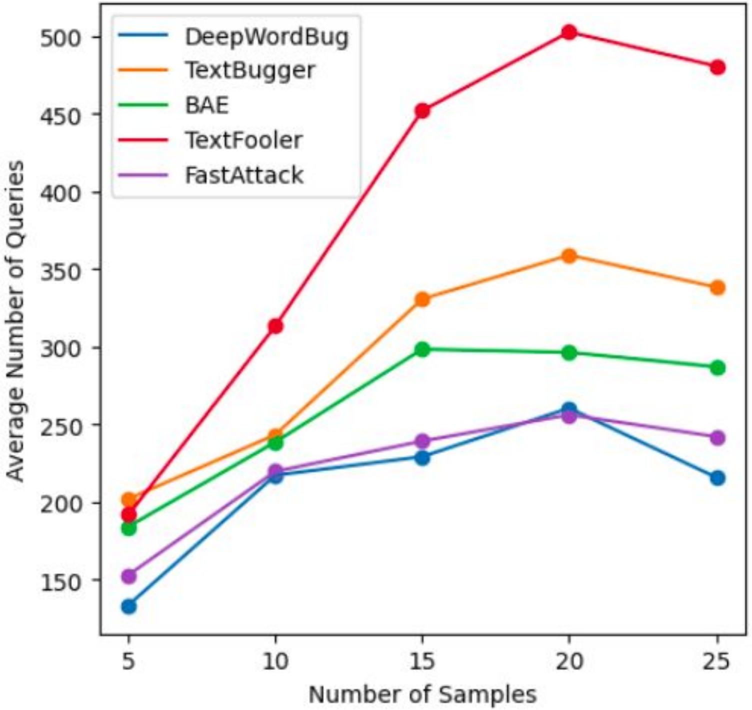
FastAttack vs BERT-Attack (Li et.al.,2020)

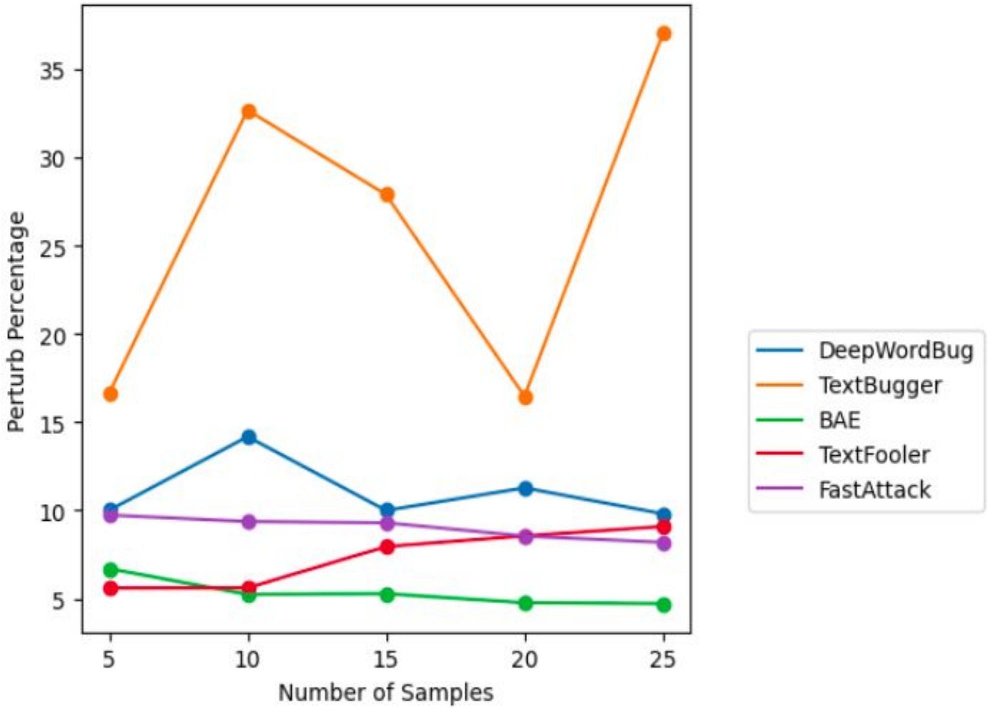
Model	Dataset	Attack method	Successful attacks (Out of 15)	Original accuracy	Attacked accuracy	Success rate	Perturbed %	Avg num queries	Attack time (in seconds)
ALBERT-base-v2	YELP-polarity	fastattack	14	78.95%	5.26%	93.33%	14.74%	**34.6**	**12.19**
		bert-attack	**15**		0.0%	**100%**	**12.52%**	110.8	123.67
	IMDB	fastattack	15	93.75%	0.0%	100%	4.45%	**352.0**	**253.93**
		bert-attack	15		0.0%	100%	**1.75%**	347.27	404.62
BERT-base-uncased	YELP-polarity	fastattack	14	78.95%	5.26%	93.33%	14.74%	**34.6**	**12.02**
		bert-attack	**15**		**0.0%**	**100%**	**12.52%**	110.8	121.36
	IMDB	fastattack	14	93.75%	6.25%	93.33%	5.34%	**365.33**	**238.02**
		bert-attack	**15**		**0.0%**	**100.0%**	**1.75%**	347.27	420.75

More comparisons



More comparisons





Summarising Results

- FastText Embeddings are high-performance embeddings, outperforming Word2Vec and GloVE embeddings in most cases
- FastAttack is a simple and highly potent attack
    - No complex transformers, no language models
    - Still gives some of the latest attacks a run for their money
    - Is indeed “fast”
    - Replacement is done with valid words only
        - Replacements are always available, even for Out-of-Vocabulary words

Member contributions

Roll No.	Member Name	Contribution
20111002	Abhishek Krishna	Trained Embeddings, Comparisons among Embeddings, FastAttack implementation, Comparisons with previous attack methods, Presentation
20111017	Deeksha Arora	Trained Embeddings, Comparisons among word Embeddings, Comparisons with previous attack methods, Report
20111044	Preeti Singh	Comparisons among Embeddings on DistilBERT-base, Report
20111054	Sambhrant Maurya	Trained Embeddings, Comparisons among embeddings, FastAttack implementation, Comparisons with previous attack methods, Presentation
20111061	Shruti Sharma	Trained Misspelling Oblivious Word Embeddings (MOE), Comparisons among Embeddings on RoBERTa-base, Report

References

[1] Ji Gao, Jack Lanchantin, Mary Lou Soffa, Yanjun Qi, “Black-box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers”, (2018)

[2] J. Li, S. Ji, T. Du, B. Li, and T. Wang, “Textbugger: Generating adversarial text against real-world applications,” Proceedings 2019 Network and Distributed System Security Symposium, 2019.

[3] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits, “Is bert really robust? a strong baseline for natural language attack on text classification and entailment,” 2020.

[4] D. Pruthi, B. Dhingra, and Z. C. Lipton, “Combating adversarial misspellings with robust word recognition,” 2019.

[5] L. Li, R. Ma, Q. Guo, X. Xue, and X. Qiu, “Bert-attack: Adversarial attack against bert using bert,” 2020.

[6] S. Garg and G. Ramakrishnan, “Bae: Bert-based adversarial examples for text classification,” 2020.

[7] V. Malik, A. Bhat, and A. Modi, “Adv-olm: Generating textual adversaries via olm,” 2021.

[8] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” 2017.

[9] S. Ren, Y. Deng, K. He, andW. Che, “Generating natural language adversarial examples through probability weighted word saliency,” 2019

[10] J. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi, “TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP,” 2020

[11] Zang et.al. (2020), Word-level Textual Adversarial Attacking as Combinatorial Optimization