GENERATING UNIVERSAL LANGUAGE ADVERSARIAL EXAMPLES BY UNDERSTANDING AND ENHANCING THE TRANSFERABILITY ACROSS NEURAL MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep neural network models are vulnerable to adversarial attacks. In many cases, malicious inputs intentionally crafted for one model can fool another model in the black-box attack setting. However, there is a lack of systematic studies on the transferability of adversarial examples and how to generate universal adversarial examples. In this paper, we systematically study the transferability of adversarial attacks for text classification models. In particular, we conduct extensive experiments to investigate how various factors, such as network architecture, input format, word embedding, and model capacity, affect the transferability of adversarial attacks. Based on these studies, we then propose universal black-box attack algorithms that can induce adversarial examples to attack almost all existing models. These universal adversarial examples reflect the defects of the learning process and the bias in the training dataset. Finally, we generalize these adversarial examples into universal word replacement rules that can be used for model diagnostics.

1 INTRODUCTION

Deep neural networks are powerful and have been widely applied in natural language processing. However, recent studies demonstrate that these models are vulnerable to adversarial examples, which are malicious inputs intentionally crafted to fool the models. Although generating adversarial examples for texts has proven to be a more challenging task than for images due to their discrete nature, a number of methods have been proposed to generate adversarial text examples and reveal the vulnerability of deep neural networks in natural language processing (NLP) tasks including reading comprehension (Jia & Liang, 2017), text classification (Samanta & Mehta, 2017; Wong, 2017; Liang et al., 2018; Alzantot et al., 2018), machine translation (Zhao et al., 2018; Ebrahimi et al., 2018; Cheng et al., 2018), dialogue systems (Cheng et al., 2019), and dependency parsing (Zheng et al., 2020). These methods attack text examples by replacing, scrambling, and erasing characters or words or other language units. To settle the susceptible attack direction, they require a large number of queries to the target model for the predictions of given inputs. Thus the adversarial examples are typically generated for a specific model. This motivates the main questions we aim to answer in this paper: *Are there universal adversarial examples that can fool almost every neural network-based model?* And *are there universal attack rules for constructing such universal adversarial examples?*

It is well known that adversarial examples exhibit black-box transferability, meaning that adversarial examples generated for one model can fool another model (Szegedy et al., 2013). Transfer attackers launch white-box attacks on local models to find candidate adversarial examples that may transfer to the target model. However, which factors most affect the transferability of adversarial examples is still unclear, especially for NLP models. In this study, we quantitatively investigate how adversarial transferability is impacted by several critical factors, including the network architecture, input form, word embedding type, and model capacity. Based on the understanding of transferability among various neural models, we study whether it is possible to craft universal, model-agnostic text adversarial examples for almost all existing models.

Universal adversarial examples have at least two advantages. First, the adversaries do not need white-box access to the target models. They launch the attacks by their own models trained on similar data, which can transfer across models (Moosavi-Dezfooli et al., 2017). Second, universal adver-

Table 1: Three adversarial examples that successfully fool all 63 models, crafted by using universal adversarial word replacement rules discovered by our proposed algorithm.

1	Senate Panel Gives NASA Extra Money (AP) AP - NASA would get #36;16.4 billion next year under a bill a Senate committee approved Tuesday, reversing a decision by House lawmakers to ~~cut~~ contract the space agency's budget below this year's levels.
2	Deal in Congress to ~~keep~~ preserve tax cuts, Widening Deficit Republican and Democratic leaders agreed to extend $5 billion worth of tax cuts sought by ~~President~~ Chairman Bush without trying to pay for them.
3	Nortel Downsizes Again Aug. 23, 2004 (TheDeal.com) Problem-plagued Nortel ~~Networks~~ Web Corp. announced plans Thursday, Aug. 19, to eliminate an additional 3,500 jobs and fire seven more senior ~~executives~~ administrators as the company labors to reinvent.

sarial examples are a useful analysis tool because, unlike typical attacks, they are model-agnostic. Thus, they highlight general input-output patterns learned by a model. We can leverage this to study the influence of dataset biases and to identify those biases that are learned by models.

In this study, we first systematically investigated a few critical factors of neural models, including network architectures (LSTM, CNN, or Transformer), input forms (character, sub-word, or word), embedding types (GloVe, word2vec, or fastText), and model capacities (various numbers of layers) and how they impact the transferability of text adversarial examples through extensive experiments on two datasets of text classification. We vary one factor at a time while fixing all others to see which factor is more significant, and found that the input form has the greatest influence on the adversarial transferability, following by network architecture, embedding type, and model capacity. Then, we propose a genetic algorithm to find an optimal ensemble with minimum number of members on the basis of our understanding of the adversarial transferability among neural models. The adversarial examples generated by attacking the ensemble found by our algorithm strongly transfer to other models, and for some models, they exhibit better transferability than those generated by attacking models with different random initialization. Finally, we generalize the adversarial examples constructed by the ensemble method into universal semantics-preserving word replacement rules that can induce adversaries on any text input strongly transferring to any neural network-based NLP model (see Table 1 for some examples). Since those rules are model-agnostic, they provide an analysis of global model behavior, and help us to identify dataset biases and to diagnose heuristics learned by the models.

2 Related Work

2.1 Transfer-based Attacks

Observing that adversarial examples often transfer across different models (Szegedy et al., 2013), the attackers run standard white-box attacks on local surrogate models to find adversarial examples that are expected to transfer to the target models. Unfortunately, such a straightforward strategy often suffers from overfitting to specific weaknesses of local models and transfer-based attacks typically have much lower success rates than optimization-based attacks directly launched on the target models. To answer this problem, many methods have been proposed to improve the transfer success rate of adversarial examples on the target models by perturbing mid-layer activations (Zhou et al., 2018; Huang et al., 2019; Inkawhich et al., 2020; Wu et al., 2020), adding regularization terms to the example generation process (Dong et al., 2018; Huang et al., 2019), or ensembling multiple local models (Wu et al., 2018; Tramèr et al., 2018; Liu et al., 2017; Wallace et al., 2019).

Ensemble-based methods are most related to this study. Liu et al. (2017) hypothesized that if an adversarial example remains adversarial for multiple models, then it is more likely to transfer to other models as well. Following this hypothesis, they improved transferability rates by using an ensemble of local models. Ensemble-based methods have been used to generate transferable adversarial examples both in computer vision (Wu et al., 2018; Tramèr et al., 2018) and NLP (Liu et al., 2017; Wallace et al., 2019). Wu et al. (2018) found that the local non-smoothness of loss surface harms the transferability of generated adversarial examples, and proposed a variance-reduced attack to enhance the transferability by applying the locally averaged gradient to reduce the local oscillation of the loss surface. Unlike the methods that ensemble the predictions of different models, more transferable adversarial examples are generated by optimizing a perturbation over an ensemble of

transformed images so that the generated examples are less sensitive to the local source models (Xie et al., 2019; Dong et al., 2019).

2.2 UNIVERSAL ADVERSARIAL EXAMPLES IN NLP

In the text domain, Wallace et al. (2019) searched for universal adversarial triggers: input-agnostic sequences of tokens that trigger a model to produce a specific prediction when concatenated to any input from a dataset. They focused on model-specific concatenated tokens generated using gradients under the white-box setting, and the founded universal triggers ("zoning tapping fiennes" for example) are usually human unreadable. Ribeiro et al. (2018) presented semantic-preserving perturbations that cause models to change their predictions by the paraphrases generated via back-translation, and generalized these perturbations into universal replacement rules that induce adversaries on many text instances. They use the word "universal" to mean that their replacement rules can be used to any input text if some rules are matched with the input, but those rules were still generalized for some specific models. In contrast, we want to find the universal adversarial replacement rules by which the crafted adversarial examples can fool almost all existing models. Besides, the number of their replacement rules is quite small, and many texts do not meet the condition specified by their rules, while our adversarial word replacement rules can be applied to most texts, leading to higher success rates on various neural NLP models.

Table 2: Adversarial transferability on AGNEWS and MR datasets with PWWS and GA attacks.

Transferability	**AGNEWS**		**MR**	
	PWWS	**GA**	**PWWS**	**GA**
Architecture	0.197	0.018	0.145	0.021
Input form	0.286	0.030	0.285	0.049
Embedding	0.085	0.015	0.114	0.015
Capacity	0.066	0.013	0.045	0.011

Table 3: Adversarial transferability among various neural network architectures.

Model	**LSTM**	**BiLSTM**	**CNN**	**BERT**
LSTM	0.448	0.394	0.353	0.190
BiLSTM	0.387	0.420	0.337	0.183
CNN	0.343	0.334	0.442	0.169
BERT	0.357	0.346	0.348	0.396

3 ADVERSARIAL TRANSFERABILITY AMONG NEURAL MODELS

We first want to investigate which factor of NLP neural models most affect the adversarial transferability by varying one factor at a time while fixing all others to see the differences in their accuracy. The factors to be investigated include network architectures (LSTM, CNN, or Transformer), input forms (character, sub-word, or word), embedding types (GloVe, word2vec, or fastText), and model capacities (various numbers of layers). Technically, we generate the adversarial examples by attacking a source model and pass the generated examples through other models for comparison.

3.1 EXPERIMENTAL SETTINGS

We use convolutional neural network (CNN), long short-term memory (LSTM), and bidirectional LSTM as base models with 1, 2, and 4 layers (an additional 6-layer one for CNN). Those networks can take three forms as input: word, character, and word + character. If word-based models are used, their word embeddings can be randomly initialized or pre-trained by GloVe (Pennington et al., 2014), word2vec (Mikolov et al., 2013), or fastText (Joulin et al., 2016). When taking word + character as input, the models are initialized with the embeddings pre-trained by ELMo (Peters et al., 2018). We also put BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), and ALBERT (Lan et al., 2019) into the model pool for analysis, and the total number of models under investigation is 63 (see Appendix A for details), which cover the popular neural networks that used in NLP literature.

All the models are investigated under two recently proposed attack algorithms, PWWS (Ren et al., 2019) and GA (Alzantot et al., 2018). We base our sets of allowed word substitutions on the synonyms created in WordNet (Miller, 1995), and for any word in a text, the word to replace must have the same part-of-speech (POS) as the original one. Ren et al. (2019) described a greedy algorithm, called Probability Weighted Word Saliency (PWWS), for text adversarial attack based on word substitutions with synonyms. The word replacement order is determined by taking both word saliency and prediction probability into account. Alzantot et al. (2018) developed a generic algorithm-based attack, denoted by GA, to generate semantically and syntactically similar adversarial examples.

Table 4: Adversarial transferability of models with different input forms and embedding types.

Input	Random	GloVe	word2vec	fastText	Character	ELMo	BERT	Average
Random	0.457	0.389	0.445	0.434	0.214	0.315	0.166	0.346
GloVe	0.481	0.503	0.489	0.493	0.219	0.336	0.174	0.385
word2vec	0.473	0.413	0.472	0.461	0.216	0.316	0.165	0.360
fastText	0.481	0.442	0.482	0.488	0.222	0.330	0.169	0.373
Character	0.261	0.233	0.256	0.256	0.386	0.300	0.186	0.268
ELMo	0.406	0.379	0.401	0.405	0.256	0.679	0.216	0.392
BERT	0.348	0.329	0.343	0.348	0.328	0.408	0.396	0.357
Average	0.415	0.384	0.413	0.412	0.263	0.383	0.210	0.354

They also use a language model (LM) to rule out candidate substitute words that do not fit within the context. However, unlike PWWS, ruling out some candidates by the LM will greatly reduce the number of candidate substitute words (65% off in average). For consistency, we report the robust accuracy under GA attack without using the LM.

We experimented on two text classification datasets: Sentiment Movie Reviews (MR) (Pang & Lee, 2005) and AG News corpus (AGNEWS) (Zhang et al., 2015). MR (Pang & Lee, 2005) has $10,000$ movie reviews for binary (positive or negative) sentiment classification, and AGNEWS (Zhang et al., 2015) consists of about $120,000$ news articles pertaining to four categories. All models are trained on the standard training set with the cross entropy loss. For each dataset, we attack $1,000$ randomly selected test examples. For evaluating their transferability on other models, we randomly choose 500 adversarial examples that successfully cause the source model to make incorrect predictions. The transferability between each possible pair of all the models is shown in Appendix B.

3.2 Significance of Various Factors

To find out which factor affects the transferability of the adversarial attack the most, we vary one factor at a time while fixing all others for each model in the pool, and compare their transferability rates. For example, we take a 2-layer word-based LSTM model randomly initialized, denoted as "LSTM-Word-Random-2", as a target model. If we want to know the impact of network architecture, we generate $1,000$ adversarial examples each by attacking BiLSTM-Word-Random-2, and CNN-Word-Random-2, and use randomly selected 500 examples each of successful attack to evaluate the accuracy of the target model. If we want to understand the influence of word embedding, the adversarial examples will be crafted by LSTM-Word-GloVe-2, LSTM-Word-word2vec-2, and LSTM-Word-fastText-2 models.

Since some models may be inherently more vulnerable than others, we want to remove this bias and accurately measure the impact of different factors on adversarial transferability. For each target model, we train two instances with the same setting but different random initialization, and obtain its *base transferability rate* by generating adversarial examples for one model and testing them on another. This base transferability rate of a model will be subtracted from all the actual transfer attack rates obtained when taking the model as target. We report the average of absolute transfer attack rates (subtracted) in Table 2. We found that the input form has the greatest influence on the adversarial transferability, followed by network architecture, pre-trained word embeddings, and model capacity in descending order no matter what attack algorithm or dataset is used.

We also observe from Table 2 that PWWS generally overfits to specific models more severely than GA. Note that the smaller the values is, the more the adversarial transferability rate is close to its base (intra-model) transferability rate. Thus, the adversarial transferability of the examples generated by GA algorithm are less sensitive to the changes in model configuration. One possible explanation is that GA introduces some randomness (e.g. mutation operation) into the generation process, which prevents the generated examples from overfitting to specific weaknesses of local models. However, it does not mean the adversarial examples created by GA largely transfer better than those crafted by PWWS because the latter produces the higher base transferability rates than the former.

3.3 Intra-Factor Transferability

Let us drill down into each specific factor. Table 3 shows adversarial transferability among different network architectures, each of them has several implementations. For example, the architecture

of "BERT" includes three variants: vanilla BERT, RoBERTa and ALBERT. Each cell (i, j) in the table reports the transferability between two classes of models i and j. The value of each cell is computed as follows: for each possible pair of models (s, t) where model s belongs to class i and model t belongs to class j, we first calculate the transferability rate between models s and j, i.e. the percentage of adversarial examples produced using model s misclassified by model t; we then average these transferability rates over all the possible pairs.

As we can see from Table 3, adversarial transfer is not symmetric, i.e. adversarial examples produced using models from class i can transfer to those from class j easily does not means the reverse is also true, which again confirms the finding of (Wu et al., 2018). As expected, intra-model adversarial example transferability rates are consistenlty higher than inter-model transferability ones. The adversarial examples generated using BERTs slightly transfer worse than other models whereas BERTs show much more robust to adversarial samples produced using the models from other classes. It is probably because BERTs were pre-trained with large-scale text data and have different input forms (i.e. sub-words). We found BERTs tend to distribute their "attention" over more words of an input text than others, which make it harder to change their predictions by perturbing just few words. In contrast, other models often "focus" on certain key words when making predictions, which make them more vulnerable to black-box transfer attacks (see Appendix C).

In Table 4, we report the impact of input forms and embedding types on the adversarial transferability. Each cell is obtained by the method as the values reported in Table 3. The pre-trained models show to be more robust against black-box transfer attacks no matter their word embeddings or other parameters or both are pre-trained with large-scale text data. Character-based models are more robust to transfer attacks than those taking words or sub-words as input, and their adversarial examples also transfer much worse than others.

3.4 Main Findings

Our major findings on the transferability among different neural models are summarized as follows:

- No matter what attack algorithm or dataset is used, the input form has the greatest impact on the adversarial transferability, followed by the architecture, word embeddings, and model capacity.
- The adversarial examples generated using BERTs slightly transfer worse than others, but BERTs show much more robust to adversarial examples produced using other models. We found that BERTs tend to distribute their "attention" over more words than others, which make it harder to change their predictions by perturbing just few words.
- Adversarial transfer is not symmetric, and the transferability rates of intra-model adversarial examples are consistenlty higher than those of inter-model ones.
- PWWS generally tends to overfit to specific local models more severely than GA algorithm since GA introduces some randomness into the generation process, which prevents its generated examples from overfitting to specific weaknesses of local models.
- Character-based models are more robust to transfer attacks than those taking words or sub-words as input, and their adversarial examples also transfer much worse than others.
- Pre-trained models show to be more robust against black-box transfer attacks no matter their word embeddings or other parameters or both are pre-trained with large-scale text data.

4 Universal Adversarial Example Generation

In this section, we attempt to find an optimal ensemble model that can be used to craft adversarial examples that strongly transfer across other models. We then distill the ensemble attack into universal word replacement rules that can be used to generate the adversarial examples with high transferability. These rules can help us to identify dataset biases and analyze global model behavior.

4.1 Ensemble Method

Given an ensemble, we can generate adversarial examples to fool the ensemble model by applying word substitution-based perturbations to input texts. We take the average of all the logits produced by the member models as final prediction of the ensemble. Observing that the transferability

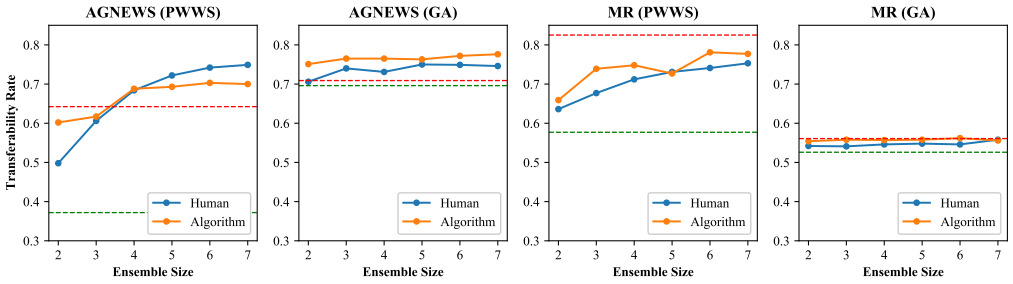

Figure 1: Transferability rates of the adversarial examples generated using different ensembles with various sizes on both AGNEWS and MR datasets under two attack algorithms (PWWS and GA). The upper red dotted line represents the average of the *base transferability rates* (defined in Section 3.2) that theoretically are highest rates that can be achieved using a single local model, and the lower green line shows the average of transferability rates over all the possible pair of models.

of adversarial examples heavily depends on the ensemble used to generate them, we propose a population-based genetic algorithm to find a close-to-optimal ensemble.

In the proposed algorithm, a candidate solution v of this problem is a set of models $(s_1, s_2, \ldots, s_m)$, where m is the pre-defined size of ensemble. A fitness function evaluates each solution to decide whether it will contribute to the next generation of solution. We define a function $r(s, t)$ as the percentage of adversarial samples produced using model s misclassified by model t. For a solution $v = (s_1, s_2, \ldots, s_m)$, the fitness function $f(x)$ that returns a measure of the candidate's fitness which we want to maximize is defined as follows:

$$f(x) = \sum_{t \in T} \{\max_{s_j \in v} [r(s_j, t)]\} \,/\, |T|, \tag{1}$$

where T is a set of all candidate models under consideration, and $|T|$ is the cardinality of set T. Let $P(k)$ define a population of candidate solutions, v_i^k, at time k: $P(k) = \{v_1^k, v_2^k, \ldots, v_n^k\}$, where n is the size of population. Initial populations $P(0)$ are selected randomly. After evaluating each candidate, the algorithm selects pairs based on fitness for recombination that uses genetic operators to produce offspring having features of their parents. We takes two candidate solutions and merge their sets, randomly selecting m models from the merged set to produce two new candidates. Mutation is another important genetic operator that takes a single candidate and randomly replaces at most one of it's models with another from T. The generation algorithm continues until the number of generations reaches the maximum value.

To evaluate the ensembles found by the population-based algorithm, we ask a senior researcher to select the ensembles as competitors. This researcher uses a simple strategy to make selection: first choose the model whose adversarial examples yield the highest transferability, and then gradually add complementary models which are different from those already in the ensemble in the aspects of input form, architecture, and embedding type. We list the ensembles selected by the algorithm or human expert in Appendix D.

In Figure 1, we show the transferability rates of the adversarial examples produced using the ensembles with various sizes on both AGNEWS and MR datasets under two attack algorithms (PWWS and GA). The reported transferability rates are averaged over all the remaining models except those used to produce the adversarial examples. We found that in most cases the adversarial examples produced using the ensembles founded by the proposed genetic algorithm transfer better across different models than those selected by human expert, especially when the ensemble size is small. The ensemble method clearly performs superior to a single model-based transfer method, and in some cases the transferability rates achieved by the ensemble method even go beyond the upper red dotted line. When the ensemble size is greater than 6, the marginal gains in average transferability rate are negligible no matter what attack algorithm or dataset is used in our experimental setting. The transferability rates of the adversarial examples generated by GA grow more slowly than the rates of those by PWWS when the size of ensemble increases.

Inputs:
$\mathcal{D}$: a set of training examples.
$\mathcal{Z}$: a set of class labels.
f: an ensemble model that outputs a logit for each class $z \in \mathcal{Z}$.
Output: a set of word replacement rules as well as their salience.
Algorithm:
for each training instance (x, y) in $\mathcal{D}$
  for each word w_i in the input text x
    for each word $\hat{w}_i$ that can be used to replace w_i
      $\hat{x}_i$ = replace w_i with $\hat{w}_i$ in x.
      $c(y, w_i \rightarrow \hat{w}_i) = c(y, w_i \rightarrow \hat{w}_i) + 1$.
      for each label $z \in \mathcal{Z}$
        if $z = y$ **then**
          $h(y, w_i \rightarrow \hat{w}_i) = h(y, w_i \rightarrow \hat{w}_i) + f(x_i; z) - f(\hat{x}_i; z)$.
        else
          $h(y, w_i \rightarrow \hat{w}_i) = h(y, w_i \rightarrow \hat{w}_i) + f(\hat{x}_i; z) - f(x_i; z)$.
for each word replacement rule
  $h(z, w \rightarrow \hat{w}) = h(z, w \rightarrow \hat{w})/c(z, w_i \rightarrow \hat{w}_i)$.

Figure 2: An algorithm to discover universal adversarial word replacement (UAWR) rules.

4.2 Mining Universal Adversarial Word Replacement Rules

We have shown in Section 4.1 that the adversarial examples generated by using the ensemble whose members are carefully selected can strongly transfer to other models. We hypothesize that if we can distill the ensemble attack into some word replacement rules, the adversarial examples crafted by applying the distilled rules to perturb input texts also can transfer well across different models. In this section, we want to discover such word replacement rules using ensemble model, and those rules are expected to be used to generate the model-agnostic examples of transferable hostility. Besides, such rules (if any) also can help us to understand and identify dataset biases "unknowingly" exploited by the models for prediction.

A word replacement rule is defined as a pair $(z, w \rightarrow \hat{w})$, where z is a class label, and $w \rightarrow \hat{w}$ means to replace the original word w with $\hat{w}$ when the gold label is z. Each rule is associated with a salience $h(z, w \rightarrow \hat{w})$ specifying the priority of the rule, and a higher number denotes a higher priority. We propose an algorithm to discover Universal Adversarial Word Replacement (UAWR) rules as shown in Figure 2. The idea behind this algorithm is to estimate the changes in log-likelihood caused by the word replacements. Once such rules are obtained, they can be used to generate adversarial examples as follows: given an input sentence x and its label y, we find a word w_i in x which has the highest value of $h(y, w_i, \hat{w}_i)$ and replace w_i with $\hat{w}_i$ in x; for all the remaining words in x we repeat the above step until the percentage of words that can be modified reach a given threshold. Note that such adversarial examples are generated without access to target models.

Table 5: Attack success rates of the adversarial examples generated by applying the word replacement rules found by our algorithm (UAWR) and pointwise mutual information (PMI) on 63 models with AGNEWS and MR datasets. "Word%" denotes the average percent of words actually modified, and "Succ%" the attack success rate in terms of the number of sentences. The maximum percentage of words that are allowed to be modified was set to 30%.

Success Rate	**AGNEWS**				**MR**			
	PMI		**UAWR**		**PMI**		**UAWR**	
	Succ%	**Word%**	**Succ%**	**Word%**	**Succ%**	**Word%**	**Succ%**	**Word%**
ALL	29.0	14.3	62.7 (+33.7)	15.0	68.0	10.6	86.8 (+18.8)	9.0
Word	31.5	14.1	69.9 (+38.4)	14.4	72.7	10.5	90.1 (+17.4)	8.4
Character	23.5	13.4	49.2 (+25.6)	15.6	61.4	10.0	80.0 (+18.6)	10.1
ELMo	29.4	16.1	57.5 (+28.1)	16.0	69.0	11.9	89.3 (+20.3)	9.5
BERT	17.9	14.2	33.4 (+15.5)	16.1	35.9	10.9	65.8 (+29.9)	11.1

Table 6: Five adversarial word replacement rules discovered from MR dataset each for the positive and negative categories as well as their changes in the pointwise mutual information (PMI).

Class	Word Substitution			PMI(Word; Positive)	PMI(Word; Negative)
Positive	funny	$\rightarrow$	laughable	$5.792 \rightarrow 0.000$ ($\mathbf{5.792}\downarrow$)	$4.980 \rightarrow 6.478$ ($\mathbf{1.499}\uparrow$)
	average	$\rightarrow$	mediocre	$5.741 \rightarrow 2.086$ ($\mathbf{3.655}\downarrow$)	$5.004 \rightarrow 6.408$ ($\mathbf{1.404}\uparrow$)
	glorious	$\rightarrow$	splendiferous	$6.478 \rightarrow 0.000$ ($\mathbf{6.478}\downarrow$)	$0.000 \rightarrow 0.000$ ($\mathbf{0.000}\uparrow$)
	giddy	$\rightarrow$	woozy	$6.063 \rightarrow 0.000$ ($\mathbf{6.063}\downarrow$)	$4.478 \rightarrow 6.478$ ($\mathbf{2.000}\uparrow$)
	brilliant	$\rightarrow$	brainy	$6.168 \rightarrow 0.000$ ($\mathbf{6.168}\downarrow$)	$4.109 \rightarrow 6.478$ ($\mathbf{2.369}\uparrow$)
Negative	toilet	$\rightarrow$	bathroom	$0.000 \rightarrow 6.478$ ($\mathbf{6.478}\uparrow$)	$6.478 \rightarrow 0.000$ ($\mathbf{6.478}\downarrow$)
	excellent	$\rightarrow$	splendid	$6.013 \rightarrow 6.478$ ($\mathbf{0.466}\uparrow$)	$4.620 \rightarrow 0.000$ ($\mathbf{4.620}\downarrow$)
	bizarre	$\rightarrow$	outlandish	$5.478 \rightarrow 6.478$ ($\mathbf{1.000}\uparrow$)	$5.478 \rightarrow 0.000$ ($\mathbf{5.478}\downarrow$)
	excruciating	$\rightarrow$	harrowing	$0.000 \rightarrow 6.256$ ($\mathbf{6.256}\uparrow$)	$6.478 \rightarrow 3.671$ ($\mathbf{2.807}\downarrow$)
	routine	$\rightarrow$	everyday	$2.230 \rightarrow 6.478$ ($\mathbf{4.248}\uparrow$)	$6.400 \rightarrow 0.000$ ($\mathbf{6.400}\downarrow$)

We report the attack success rates of the adversarial examples generated by applying UAWR rules in Table 5. As a reference, PWWS algorithm achieved 66.6% and 90.8% success rates in average on AGNEWS and MR datasets respectively under the black-box setting. The attacks based on UAWR rules are comparable to the algorithm that requires a large number of queries to the target model. Although the ensembles founded by our genetic algorithm can produce the adversarial examples with more transferability than those selected by human expert, the algorithm will yield different ensembles with different datasets or attack methods. Therefore, we use the ensemble consisting of six models (see Appendix D) selected by human expert to discover these UAWR rules in this experiment. We listed five adversarial word replacement rules each for the positive and negative categories discovered from MR dataset in Table 6. It can be seen from the differences in the pointwise mutual information (PMI) between words and classes before and after replacements that those words clearly can be identified as the dataset biases. The PMI of a pair of discrete random variables quantifies the discrepancy between the probability of their coincidence given their joint distribution and their individual distributions. In this case, it is used to find collocations and associations between words and labels, and the PMI of a word w and a label $z \in \mathcal{Z}$ can be computed as $\text{PMI}(w, z) = p(w, z)/p(w)p(z)$, where $p(\cdot)$ assigns a probability to each possible value.

We obtain similar word replacement rules by ranking all the hypothesis words according to their PMI with each label from a training set. For each label $z \in \mathcal{Z}$ and a possible word replacement $w \rightarrow \hat{w}$, the similar salience of $h(z, w \rightarrow \hat{w})$ can be computed as follows:

$$h(z, w \rightarrow \hat{w}) = [\text{PMI}(w, z) - \text{PMI}(\hat{w}, z)] + \sum_{z' \in \mathcal{Z}, z' \neq z} [\text{PMI}(\hat{w}, z') - \text{PMI}(w, z')] \tag{2}$$

We also report the attack success rates of the adversarial examples generated by the word replacement rules obtained using PMI only in Table 5. The adversarial examples produced by UAWR rules achieved stronger transferability than those by PMI rules. We believe that it is because UAWR rules are distilled using the logits predicted by models, and the changes in the logits reflect both the characteristics of neural networks and the contexts in which those word replacements are applied.

5 Conclusion

In this study, we investigated four critical factors of NLP neural models, including network architectures, input forms, embedding types, and model capacities and how they impact the transferability of text adversarial examples with 63 different models. Based on the understanding of the transferability among those models, we proposed a genetic algorithm to find an optimal ensemble of very few models that can be used to generate adversarial examples that transfer well to all the other models. We also described a algorithm to discover universal adversarial word replacement rules that can be applied to craft adversarial examples with strong transferability across various neural models without access to any of them. Finally, since those adversarial examples are model-agnostic, they provide an analysis of global model behavior and help to identify dataset biases.

REFERENCES

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018.

Minhao Cheng, Jinfeng Yi, Huan Zhang, Pin-Yu Chen, and Cho-Jui Hsieh. Seq2Sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. *Computing Research Repository*, arXiv: 1803.01128, 2018.

Minhao Cheng, Wei Wei, and Cho-Jui Hsieh. Evaluating and enhancing the robustness of dialogue systems: A case study on a negotiation agent. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4312–4321, 2019.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2018.

Qian Huang, Isay Katsman, Horace He, Zeqi Gu, Serge belongie, and Ser-Nam Lim. Enhancing adversarial example transferability with an intermediate level attack. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

Nathan Inkawhich, Kevin J. Liang, Lawrence Carin, and Yiran Chen. Transferable perturbations of deep feature distributions-feature space. In *Proceedings of the Conference of the International Conference on Learning Representations*, 2020.

Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2017.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext.zip: Compressing text classification models. *Computing Research Repository*, arXiv: 1612.03651, 2016.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *Computing Research Repository*, arXiv: 1909.11942, 2019.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. Deep text classification can be fooled. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2018.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *Proceedings of the Conference of the International Conference on Learning Representations*, 2017.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized bert pretraining approach. *Computing Research Repository*, arXiv: 1907.11692, 2019.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *Computing Research Repository*, arXiv: 1301.3781, 2013.

George A. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv preprint cs/0506075*, 2005.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *Computing Research Repository*, arXiv: 1802.05365, 2018.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2019.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2018.

Suranjana Samanta and Sameep Mehta. Towards crafting text adversarial samples. *arXiv preprint arXiv:1707.02812*, 2017.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *Proceedings of the Conference of the International Conference on Learning Representations*, 2018.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing nlp. *arXiv preprint arXiv:1908.07125*, 2019.

Catherine Wong. DANCin SEQ2SEQ: Fooling text classifiers with adversarial text example generation. *Computing Research Repository*, arXiv: 1712.05419, 2017.

Lei Wu, Zhanxing Zhu, Cheng Tai, and Weinan E. Understanding and enhancing the transferability of adversarial examples. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, 2018.

Weibin Wu, Yuxin Su, Xixian Chen, Shenglin Zhao, Irwin King, Michael R. Lyu, and Yu-Wing Tai. Boosting the transferability of adversarial samples via attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2730–2739, 2019.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Proceedings of the Conference on Neural Information Processing Systems*, 2015.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. In *Proceedings of the International Conference on Learning Representations*, 2018.

Xiaoqing Zheng, Jiehang Zeng, Yi Zhou, Cho-Jui Hsieh, Minhao Cheng, and Xuanjing Huang. Evaluating and enhancing the robustness of neural network-based dependency parsing models with adversarial examples. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2020.

Wen Zhou, Xin Hou, Yongjun Chen, Mengyun Tang, Xiangqi Huang, Xiang Gan, and Yong Yang. Transferable adversarial perturbations. In *Proceedings of the European Conference on Computer Vision*, 2018.

APPENDIX

A ALL NEURAL MODELS UNDER INVESTIGATION

We systematically investigate a few critical factors of neural models, including network architectures (LSTM, BiLSTM, CNN, or BERT), input forms (Word, character, or word + character, denoted by "W", "C", "WC" respectively), word embeddings (randomly-initialized, GloVe, word2vec, or fastText), and model capacities (various numbers of layers). All models under consideration are listed in Table 7, and we think that they cover the popular neural networks that used in NLP literature.

Table 7: All neural models under investigation.

ID	Model	ID	Model	ID	Model
[1]	LSTM-W-Random-1	[22]	BiLSTM-W-fastText-1	[43]	CNN-W-Random-2
[2]	LSTM-W-GloVe-1	[23]	BiLSTM-C-Random-1	[44]	CNN-W-GloVe-2
[3]	LSTM-W-W2vec-1	[24]	BiLSTM-WC-ELMo-1	[45]	CNN-W-W2vec-2
[4]	LSTM-W-fastText-1	[25]	BiLSTM-W-Random-2	[46]	CNN-W-fastText-2
[5]	LSTM-C-Random-1	[26]	BiLSTM-W-GloVe-2	[47]	CNN-C-Random-2
[6]	LSTM-WC-ELMo-1	[27]	BiLSTM-W-W2vec-2	[48]	CNN-WC-ELMo-2
[7]	LSTM-W-Random-2	[28]	BiLSTM-W-fastText-2	[49]	CNN-W-Random-4
[8]	LSTM-W-GloVe-2	[29]	BiLSTM-C-Random-2	[50]	CNN-W-GloVe-4
[9]	LSTM-W-W2vec-2	[30]	BiLSTM-WC-ELMo-2	[51]	CNN-W-W2vec-4
[10]	LSTM-W-fastText-2	[31]	BiLSTM-W-Random-4	[52]	CNN-W-fastText-4
[11]	LSTM-C-Random-2	[32]	BiLSTM-W-GloVe-4	[53]	CNN-C-Random-4
[12]	LSTM-WC-ELMo-2	[33]	BiLSTM-W-W2vec-4	[54]	CNN-WC-ELMo-4
[13]	LSTM-W-Random-4	[34]	BiLSTM-W-fastText-4	[55]	CNN-W-Random-6
[14]	LSTM-W-GloVe-4	[35]	BiLSTM-C-Random-4	[56]	CNN-W-GloVe-6
[15]	LSTM-W-W2vec-4	[36]	BiLSTM-WC-ELMo-4	[57]	CNN-W-W2vec-6
[16]	LSTM-W-fastText-4	[37]	CNN-W-Random-1	[58]	CNN-W-fastText-6
[17]	LSTM-C-Random-4	[38]	CNN-W-GloVe-1	[59]	CNN-C-Random-6
[18]	LSTM-WC-ELMo-4	[39]	CNN-W-W2vec-1	[60]	CNN-WC-ELMo-6
[19]	BiLSTM-W-Random-1	[40]	CNN-W-fastText-1	[61]	BERT
[20]	BiLSTM-W-GloVe-1	[41]	CNN-C-Random-1	[62]	RoBERTa
[21]	BiLSTM-W-W2vec-1	[42]	CNN-WC-ELMo-1	[63]	ALBERT

B TRANSFERABILITY AMONG DIFFERENT NEURAL MODELS

We show in Figure 3 the transferability among various neural models. The column or row headers indicate the IDs of source and target models respectively. The mapping of IDs and their models is shown in Figure 7. We generate adversarial examples by attacking a source model, and report their transferability rates by testing them on the other target models.

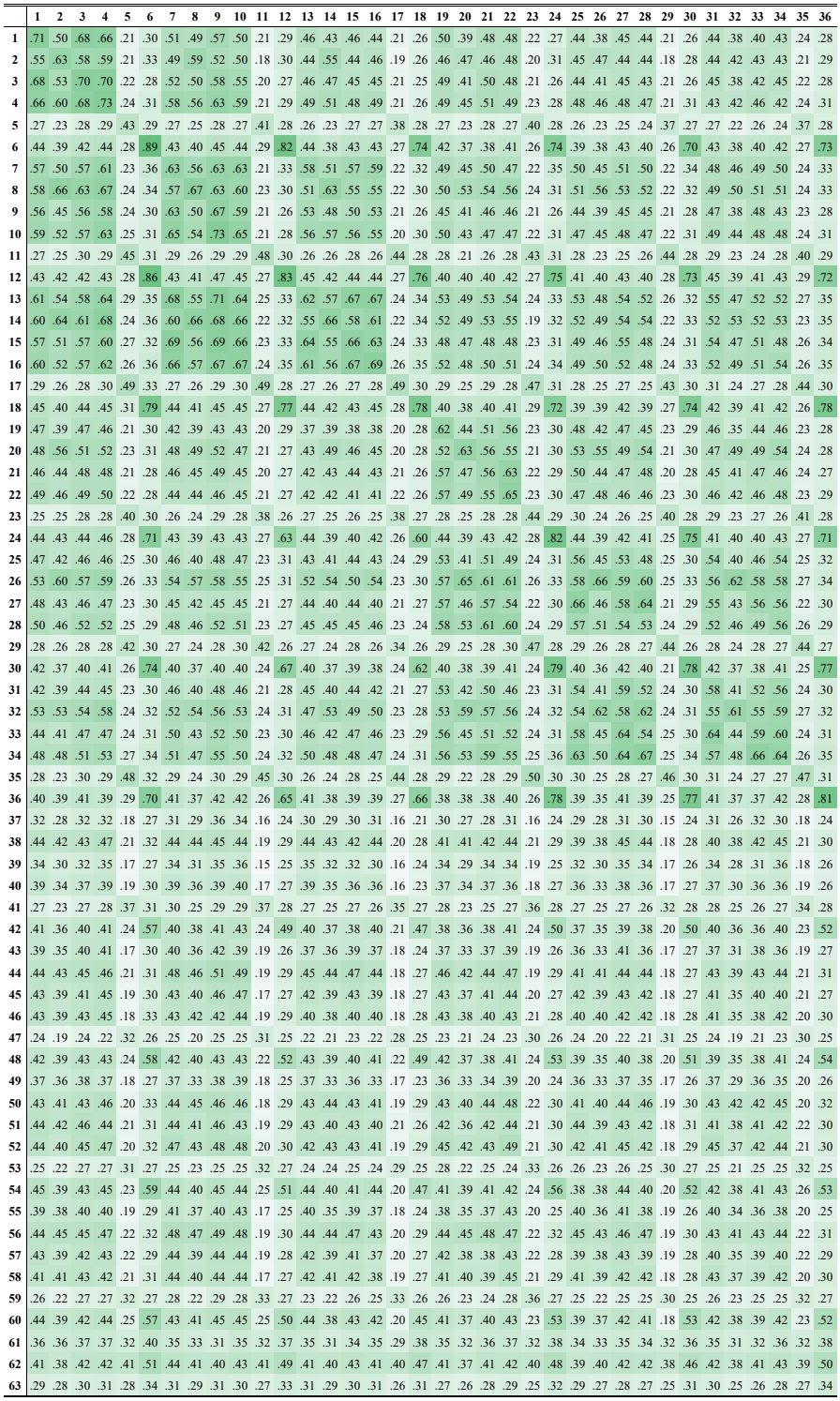

Figure 3: Transferability among neural models (part 1).

	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
1	.36	.34	.35	.35	.22	.35	.38	.33	.36	.36	.23	.33	.37	.31	.36	.36	.20	.35	.35	.31	.35	.33	.24	.33	.12	.18	.21
2	.34	.33	.35	.36	.20	.33	.39	.34	.34	.32	.23	.35	.37	.32	.33	.32	.20	.36	.36	.30	.34	.33	.20	.36	.12	.16	.22
3	.38	.39	.39	.38	.23	.32	.40	.35	.37	.35	.23	.33	.37	.34	.37	.35	.19	.35	.36	.34	.35	.34	.23	.33	.12	.16	.20
4	.40	.37	.37	.39	.22	.34	.37	.36	.38	.36	.21	.35	.39	.34	.38	.36	.22	.36	.37	.35	.36	.37	.22	.36	.12	.18	.23
5	.24	.23	.25	.26	.33	.31	.24	.23	.25	.23	.31	.31	.25	.20	.23	.24	.33	.34	.22	.21	.23	.23	.34	.32	.14	.18	.22
6	.39	.36	.37	.40	.28	.62	.37	.35	.37	.36	.30	.68	.38	.35	.36	.37	.27	.67	.38	.35	.39	.37	.29	.67	.17	.23	.30
7	.43	.40	.43	.40	.24	.37	.42	.39	.41	.38	.22	.39	.42	.39	.40	.39	.21	.41	.40	.36	.38	.39	.23	.39	.14	.17	.23
8	.41	.41	.40	.41	.24	.36	.43	.41	.39	.42	.23	.36	.41	.35	.39	.40	.20	.39	.42	.38	.40	.41	.23	.37	.14	.16	.23
9	.35	.38	.36	.38	.23	.32	.41	.35	.39	.35	.23	.34	.39	.35	.33	.36	.20	.36	.38	.33	.36	.36	.23	.33	.12	.18	.21
10	.41	.41	.40	.42	.22	.37	.40	.40	.41	.40	.24	.35	.42	.37	.37	.41	.22	.37	.39	.39	.40	.40	.23	.37	.13	.17	.21
11	.26	.25	.25	.26	.38	.34	.26	.23	.24	.22	.35	.35	.26	.24	.22	.23	.33	.37	.24	.22	.23	.24	.35	.36	.15	.23	.22
12	.38	.36	.38	.38	.30	.59	.38	.34	.37	.37	.30	.64	.38	.36	.36	.37	.27	.66	.37	.34	.37	.36	.27	.65	.18	.23	.31
13	.40	.44	.44	.45	.25	.38	.45	.43	.43	.44	.24	.41	.45	.39	.41	.43	.23	.42	.43	.40	.42	.42	.25	.40	.13	.20	.25
14	.41	.43	.41	.40	.22	.36	.44	.43	.45	.42	.23	.38	.44	.40	.41	.42	.20	.40	.43	.41	.42	.43	.23	.37	.12	.18	.21
15	.45	.46	.44	.47	.24	.40	.46	.43	.45	.43	.24	.38	.45	.40	.42	.43	.23	.41	.44	.43	.42	.44	.26	.38	.13	.19	.25
16	.41	.43	.44	.44	.27	.39	.44	.41	.46	.44	.26	.40	.46	.40	.41	.42	.24	.40	.41	.39	.41	.41	.27	.38	.13	.20	.25
17	.25	.25	.26	.25	.42	.34	.27	.25	.25	.23	.40	.36	.28	.23	.24	.25	.38	.36	.25	.24	.24	.26	.42	.35	.15	.19	.24
18	.35	.36	.36	.38	.30	.59	.37	.36	.38	.36	.29	.65	.37	.35	.36	.35	.26	.66	.36	.35	.39	.37	.27	.65	.21	.25	.31
19	.37	.33	.33	.35	.22	.33	.33	.33	.33	.32	.22	.34	.36	.29	.34	.32	.20	.35	.32	.30	.31	.32	.21	.34	.13	.19	.19
20	.36	.35	.36	.36	.22	.36	.39	.35	.37	.35	.23	.36	.37	.33	.37	.36	.20	.38	.36	.36	.36	.35	.24	.35	.14	.17	.21
21	.35	.35	.33	.34	.23	.31	.36	.35	.33	.32	.23	.34	.35	.30	.34	.33	.21	.34	.33	.31	.34	.32	.22	.34	.12	.17	.19
22	.37	.36	.35	.37	.23	.31	.37	.36	.37	.35	.24	.35	.37	.35	.36	.34	.22	.36	.39	.35	.35	.32	.22	.34	.12	.18	.19
23	.24	.25	.25	.25	.35	.32	.25	.24	.25	.23	.33	.32	.25	.21	.23	.23	.32	.33	.23	.22	.23	.24	.32	.33	.13	.19	.21
24	.36	.36	.35	.37	.27	.56	.37	.35	.36	.36	.25	.63	.37	.34	.36	.37	.27	.62	.35	.36	.36	.35	.26	.63	.16	.23	.26
25	.36	.37	.38	.37	.23	.35	.38	.37	.38	.34	.25	.37	.38	.33	.34	.33	.23	.38	.37	.33	.34	.34	.24	.37	.13	.19	.20
26	.41	.41	.39	.43	.23	.35	.43	.41	.43	.43	.26	.36	.44	.38	.41	.44	.22	.38	.45	.40	.43	.42	.26	.37	.14	.20	.23
27	.36	.35	.38	.39	.24	.34	.38	.35	.40	.36	.23	.36	.38	.32	.35	.35	.23	.37	.37	.35	.36	.35	.23	.34	.12	.17	.20
28	.40	.37	.38	.40	.25	.33	.39	.36	.40	.39	.27	.34	.40	.35	.38	.37	.24	.36	.40	.35	.38	.38	.25	.35	.11	.18	.20
29	.25	.25	.26	.27	.33	.30	.23	.23	.25	.24	.33	.30	.26	.23	.26	.23	.32	.31	.24	.23	.22	.26	.34	.30	.15	.20	.22
30	.34	.33	.34	.34	.27	.53	.36	.31	.35	.34	.25	.63	.36	.32	.34	.34	.25	.64	.33	.33	.34	.32	.24	.62	.17	.22	.25
31	.34	.35	.32	.35	.23	.34	.37	.34	.35	.35	.25	.35	.35	.32	.34	.33	.22	.35	.34	.31	.32	.33	.24	.35	.12	.18	.21
32	.39	.40	.38	.37	.23	.35	.45	.40	.41	.38	.24	.36	.44	.36	.40	.39	.22	.38	.41	.37	.39	.37	.25	.36	.13	.18	.24
33	.39	.37	.40	.38	.25	.36	.41	.35	.40	.37	.25	.36	.40	.35	.37	.37	.23	.36	.39	.34	.37	.37	.24	.36	.13	.19	.22
34	.39	.38	.39	.40	.24	.38	.42	.38	.39	.36	.26	.39	.41	.37	.38	.37	.24	.39	.40	.37	.40	.38	.24	.39	.12	.20	.23
35	.25	.25	.25	.27	.38	.34	.26	.23	.25	.23	.38	.35	.27	.26	.24	.25	.35	.35	.23	.22	.23	.26	.34	.35	.17	.22	.24
36	.36	.35	.36	.36	.26	.54	.36	.35	.35	.37	.27	.62	.39	.35	.36	.36	.26	.64	.35	.34	.37	.35	.25	.63	.18	.22	.26
37	.62	.45	.53	.50	.19	.32	.42	.35	.44	.38	.19	.34	.44	.33	.38	.38	.18	.36	.42	.33	.41	.35	.20	.33	.10	.15	.17
38	.65	.77	.66	.65	.23	.39	.55	.63	.61	.55	.22	.40	.55	.57	.56	.56	.20	.42	.57	.60	.54	.53	.21	.42	.12	.17	.22
39	.64	.52	.63	.58	.20	.35	.48	.39	.49	.44	.18	.35	.46	.37	.44	.43	.17	.38	.45	.39	.45	.42	.19	.34	.11	.15	.19
40	.65	.58	.62	.69	.20	.38	.51	.47	.55	.51	.20	.37	.48	.42	.50	.48	.19	.39	.51	.45	.48	.50	.21	.37	.10	.15	.22
41	.26	.24	.28	.27	.46	.34	.28	.23	.27	.26	.46	.37	.26	.23	.25	.24	.42	.37	.25	.23	.24	.25	.43	.34	.13	.19	.21
42	.42	.42	.42	.43	.25	.84	.41	.40	.41	.39	.25	.86	.40	.37	.38	.39	.26	.81	.37	.38	.38	.37	.26	.85	.16	.19	.25
43	.55	.47	.51	.50	.21	.35	.65	.44	.60	.50	.21	.35	.64	.42	.56	.49	.21	.36	.60	.42	.54	.47	.20	.37	.12	.15	.20
44	.54	.66	.56	.58	.23	.38	.60	.71	.62	.57	.21	.40	.60	.63	.57	.61	.20	.42	.55	.63	.57	.57	.21	.39	.12	.15	.23
45	.57	.52	.56	.57	.22	.36	.69	.53	.68	.61	.21	.37	.66	.47	.66	.59	.22	.39	.61	.49	.66	.57	.22	.40	.12	.15	.20
46	.56	.53	.54	.59	.21	.38	.65	.54	.64	.65	.22	.37	.64	.50	.64	.68	.23	.40	.61	.53	.61	.62	.24	.40	.13	.16	.19
47	.22	.21	.22	.22	.42	.30	.23	.21	.22	.21	.43	.32	.24	.21	.20	.21	.40	.32	.22	.21	.20	.22	.40	.30	.13	.18	.20
48	.42	.41	.40	.44	.25	.77	.44	.41	.45	.44	.24	.89	.43	.39	.42	.41	.24	.88	.39	.39	.42	.41	.25	.89	.16	.19	.22
49	.50	.44	.49	.49	.21	.34	.65	.45	.61	.50	.21	.34	.64	.41	.57	.50	.19	.35	.61	.42	.57	.48	.21	.35	.12	.15	.18
50	.54	.64	.54	.55	.23	.39	.56	.71	.61	.58	.25	.38	.61	.66	.58	.58	.22	.41	.57	.66	.58	.58	.23	.41	.11	.17	.24
51	.56	.54	.58	.56	.23	.39	.68	.54	.69	.63	.22	.39	.69	.51	.67	.64	.22	.41	.64	.52	.67	.61	.22	.42	.12	.15	.21
52	.55	.55	.53	.62	.23	.38	.64	.61	.69	.68	.23	.41	.65	.55	.69	.74	.22	.43	.63	.56	.66	.67	.24	.43	.14	.17	.19
53	.24	.25	.25	.24	.40	.32	.25	.25	.25	.25	.40	.33	.25	.23	.23	.24	.41	.32	.24	.22	.22	.25	.44	.32	.13	.19	.21
54	.43	.43	.41	.41	.25	.75	.44	.42	.45	.42	.24	.91	.46	.40	.42	.43	.24	.89	.39	.41	.41	.40	.24	.90	.15	.18	.22
55	.57	.47	.52	.54	.21	.36	.63	.47	.62	.53	.21	.36	.68	.43	.58	.54	.20	.38	.63	.45	.60	.52	.23	.37	.12	.16	.18
56	.59	.70	.57	.59	.22	.39	.60	.74	.62	.59	.23	.40	.58	.67	.58	.63	.22	.43	.58	.72	.60	.57	.23	.42	.12	.17	.24
57	.56	.51	.57	.59	.25	.39	.67	.53	.67	.59	.23	.38	.66	.51	.66	.64	.23	.40	.60	.49	.64	.59	.23	.41	.13	.16	.21
58	.56	.57	.55	.59	.21	.39	.64	.55	.69	.66	.21	.40	.64	.56	.64	.69	.20	.43	.61	.57	.66	.68	.23	.43	.13	.18	.21
59	.25	.25	.25	.27	.44	.33	.24	.23	.24	.22	.43	.33	.24	.23	.24	.24	.43	.34	.23	.23	.22	.25	.43	.33	.13	.19	.23
60	.45	.43	.42	.42	.26	.75	.46	.44	.47	.44	.24	.90	.46	.39	.43	.43	.25	.88	.41	.42	.41	.41	.25	.89	.15	.20	.25
61	.35	.34	.31	.33	.32	.39	.34	.33	.32	.33	.33	.40	.34	.33	.32	.32	.33	.41	.34	.33	.34	.35	.32	.40	.49	.33	.39
62	.43	.38	.42	.41	.39	.50	.39	.37	.40	.39	.38	.51	.39	.37	.40	.38	.39	.49	.39	.38	.40	.40	.40	.51	.33	.51	.49
63	.30	.29	.28	.29	.28	.37	.28	.27	.28	.27	.28	.38	.27	.26	.28	.27	.26	.38	.27	.27	.26	.27	.28	.36	.21	.26	.55

Figure 3: Transferability among neural models (part 2).

C HEATMAP OF WORD IMPORTANCE

We study the behavior of different models via word importance, which is defined as follows:

- For an original word, its importance is calculated as the difference between the log likelihood of a gold label before and after the original word is replaced with a special "unknown" symbol (<unk>)".
- For a substitute word, its importance is estimated as the difference between the log likelihood of a gold label before and after the substitute word is used to replace the original one.

Figure 5 and 6 show the importance of original and substitute words for different models. We here only consider one-layer models and take the following sentence as an example input:

Storage, servers bruise HP earnings update Earnings per share rise compared with a year ago, but company misses analysts' expectations by a long shot.

We observed that different models behave similarly: for the original words, most models mainly focus on three words, namely "Storage", "servers" and "HP"; for the substitute words, the attentions have been given to the word "depot" for most models. Thanks to such similarity in their behavior, it is possible to generate the adversarial examples using one model, which also transfer well the other models.

However, we found that the above observation is not applied to BERTs and character-based models. For the models from BERT family, they tend to distribute their "attention" over more words both for original words and substitute words. For character-based models, they also distribute their attention in the way that is clearly different from the other models. These differences can explain the poor transferability achieved by the adversarial examples generated by using BERTs and character-based models.

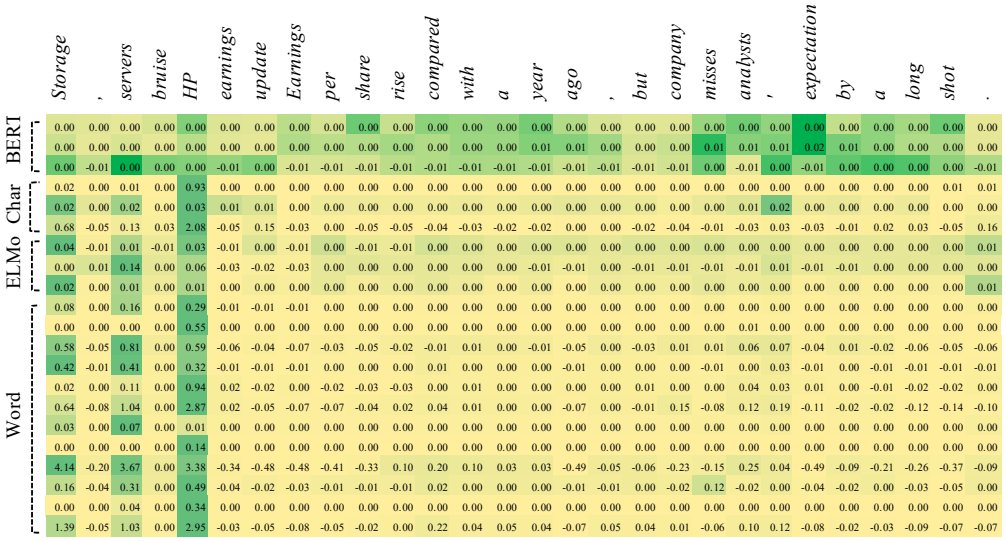

Figure 5: Importance of original words.

D THE ENSEMBLES SELECTED BY ALGORITHM OR HUMAN EXPERT

Table 8 shows ensemble models selected by the proposed genetic algorithm and human expert.

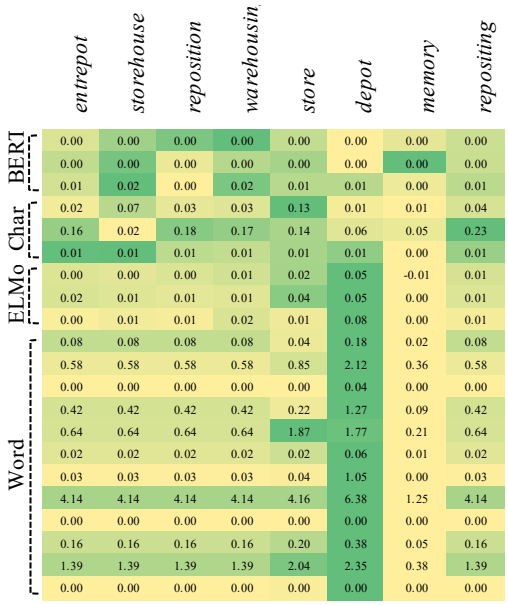

Figure 6: Importance of the words used to replace an original word "Storage" (the first word in the sentence).

Table 8: Different ensembles selected by human expert and algorithm.

Human	2	LSTM-W-Random-1, CNN-C-Random-1
	3	LSTM-W-Random-1, CNN-C-Random-1, LSTM-WC-ELMo-1
	4	LSTM-W-Random-1, CNN-C-Random-1, LSTM-WC-ELMo-1, BERT
	5	LSTM-W-Random-1, CNN-C-Random-1, LSTM-WC-ELMo-1, BERT, CNN-W-Random-1
	6	LSTM-W-Random-1, CNN-C-Random-1, LSTM-WC-ELMo-1, BERT, CNN-W-Random-1, LSTM-C-Random-1
	7	LSTM-W-Random-1, CNN-C-Random-1, LSTM-WC-ELMo-1, BERT, CNN-W-Random-1, LSTM-C-Random-1, CNN-WC-ELMo-1
Algorithm		**AGNEWS (PWWS)**
	2	LSTM-WC-ELMo-4, CNN-W-GloVe-6
	3	BiLSTM-W-GloVe-2, LSTM-WC-ELMo-4, CNN-W-fastText-4
	4	LSTM-WC-ELMo-1, BiLSTM-W-GloVe-2, CNN-W-fastText-4, RoBERTa
	5	LSTM-WC-ELMo-1, BiLSTM-W-GloVe-2, LSTM-W-Random-4, CNN-W-fastText-4, RoBERTa
	6	LSTM-WC-ELMo-2, BiLSTM-W-GloVe-2, LSTM-W-Random-4, CNN-W-fastText-4, CNN-WC-ELMo-4, RoBERTa
	7	LSTM-WC-ELMo-2, BiLSTM-W-GloVe-2, LSTM-W-Random-4, CNN-W-fastText-4, CNN-WC-ELMo-4, CNN-W-GloVe-6, RoBERTa
		AGNEWS (GA)
	2	LSTM-WC-ELMo-1, CNN-WC-ELMo-1
	3	LSTM-WC-ELMo-1, CNN-C-Random-1, CNN-WC-ELMo-1
	4	BiLSTM-WC-ELMo-4, CNN-C-Random-1, CNN-WC-ELMo-1, CNN-WC-ELMo-6
	5	BiLSTM-WC-ELMo-4, CNN-C-Random-1, CNN-WC-ELMo-1, CNN-WC-ELMo-6, RoBERTa
	6	LSTM-WC-ELMo-4, BiLSTM-WC-ELMo-4, CNN-C-Random-1, CNN-WC-ELMo-1, CNN-WC-ELMo-6, RoBERTa
	7	LSTM-WC-ELMo-4, BiLSTM-WC-ELMo-4, CNN-C-Random-1, CNN-WC-ELMo-1, CNN-WC-ELMo-2, CNN-WC-ELMo-6, RoBERTa
		MR (PWWS)
	2	LSTM-C-Random-4, CNN-WC-ELMo-6
	3	LSTM-C-Random-4, BiLSTM-W-GloVe-4, CNN-WC-ELMo-2
	4	LSTM-C-Random-4, BiLSTM-W-GloVe-4, CNN-W-fastText-2, CNN-WC-ELMo-2
	5	LSTM-W-word2vec-1, LSTM-C-Random-4, CNN-W-fastText-2, CNN-WC-ELMo-2, CNN-W-GloVe-6
	6	LSTM-W-word2vec-1, LSTM-C-Random-4, CNN-W-fastText-2, CNN-WC-ELMo-2, CNN-W-GloVe-6, RoBERTa
	7	LSTM-W-word2vec-1, LSTM-C-Random-4, BiLSTM-W-GloVe-4, CNN-WC-ELMo-2, CNN-W-GloVe-6, CNN-W-word2vec-6, RoBERTa
		MR (GA)
	2	BiLSTM-W-GloVe-4, RoBERTa
	3	LSTM-C-Random-4, BiLSTM-W-GloVe-4, RoBERTa
	4	LSTM-W-Random-1, LSTM-C-Random-4, CNN-W-GloVe-1, RoBERTa
	5	LSTM-W-Random-1, LSTM-C-Random-4, BiLSTM-W-GloVe-4, CNN-W-GloVe-1, RoBERTa
	6	LSTM-W-Random-1, LSTM-C-Random-4, BiLSTM-W-GloVe-4, BiLSTM-C-Random-4, CNN-W-GloVe-1, RoBERTa
	7	LSTM-W-Random-1, LSTM-W-word2vec-1, LSTM-C-Random-4, BiLSTM-W-GloVe-4, BiLSTM-C-Random-4, CNN-W-GloVe-1,RoBERTa

E ATTACK SUCCESS RATE USING UAWR RULES

Table 9 shows attack success rate of the adversarial examples crafted by UAWR rules under different ensemble size. As the ensemble size increases, the attack success rate is generally on the rise. Even if the ensemble size is small, the success rate can still reach 50%.

Table 10 shows the attack success rate against various maximum percentage of words that allowed to be modified.

Table 9: Attack success rate of the adversarial examples crafted by UAWR rules produced using the ensembles with different sizes.

		AGNEWS						MR					
Ensemble Size		**2**	**3**	**4**	**5**	**6**	**7**	**2**	**3**	**4**	**5**	**6**	**7**
Succ%	**All**	52.9	56.9	56.9	62.4	62.7	**63.2**	85.3	86.2	85.8	86.3	86.8	**87.0**
	Word	57.1	60.9	60.9	**71.0**	69.9	69.9	88.9	89.6	89.0	**90.3**	90.1	**90.3**
	Character	50.0	50.5	**50.1**	43.8	49.2	49.4	79.0	79.2	78.1	76.5	80.0	**80.1**
	ELMo	47.2	55.3	55.4	57.6	57.5	**60.2**	87.5	89.4	89.1	89.6	89.3	**89.8**
	BERT	27.4	30.5	33.3	32.6	**33.4**	34.3	60.0	62.5	**66.2**	63.8	65.8	**66.2**
Word%	**All**	15.1	14.9	15.1	15.0	14.9	14.9	9.3	9.3	9.2	9.1	9.0	9.0
	Word	14.6	14.4	14.7	14.5	14.5	14.5	8.6	8.6	8.6	8.4	8.4	8.4
	Character	15.6	15.7	15.8	15.7	15.6	15.6	10.2	10.2	10.1	10.2	10.1	10.1
	ELMo	16.2	15.8	15.9	16.2	16.0	15.9	10.0	9.7	9.7	9.6	9.5	9.4
	BERT	15.9	16.1	16.0	15.9	16.1	16.1	11.3	11.6	10.9	11.1	11.1	11.0

Table 10: Attack success rate against maximum percentage of words that are allowed to be modified. "Word%" denotes the maximum percentage of modified words.

Word%	**AGNEWS**	**MR**
5%	16.3	36.2
10%	27.6	54.3
15%	38.8	66.1
20%	48.8	75.3
25%	57.3	81.1
30%	63.2	87.0