From Hero to Zéroe:
A Benchmark of Low-Level Adversarial Attacks

Steffen Eger and Yannik Benz
Computer Science Department
Technische Universität Darmstadt, Germany
eger@aiphes.tu-darmstadt.de, yannik.benz@stud.tu-darmstadt.de

Abstract

Adversarial attacks are label-preserving modifications to inputs of machine learning classifiers designed to fool machines but not humans. Natural Language Processing (NLP) has mostly focused on high-level attack scenarios such as paraphrasing input texts. We argue that these are less realistic in typical application scenarios such as in social media, and instead focus on *low-level* attacks on the character-level. Guided by human cognitive abilities and human robustness, we propose the first large-scale catalogue and benchmark of low-level adversarial attacks, which we dub *Zéroe*, encompassing nine different attack modes including visual and phonetic adversaries. We show that *RoBERTa*, NLP's current workhorse, fails on our attacks. Our dataset provides a benchmark for testing robustness of future more human-like NLP models.

1 Introduction

Adversarial examples are label-preserving modifications to inputs of machine learning architectures. Their typical characteristic is that they cause little damage to humans but may maximally affect classifier performance, exposing their weaknesses and outlining the differences between human and machine text processing (Szegedy et al., 2014; Goodfellow et al., 2014; Eger et al., 2019).

While in computer vision, pixel-level attacks, which go unnoticed by humans, may lead to catastrophic failure, attacks in NLP are more challenging. Some attacks in NLP replace individual words by synonyms or hyponyms (Alzantot et al., 2018) or paraphrase whole sentences (Ribeiro et al., 2018). However, such *high-level* attacks are not only more difficult to compute (requiring available resources such as dictionaries or word embeddings) but they are also implausible in real-world scenarios such as spamming or posting in *social media*, as users would need to know the training data and/or the inner workings of the machine learning models in order to identify candidate substitutions (or have unrestrained access to model predictions). In contrast, such users would typically use *low-level* attacks on characters, such as inserting placeholder symbols (e.g., underscores), mistyping words (e.g., *Hilter* for *Hitler*), or using phonetically similar sounding words (Tagg, 2011) to fool online detection models. To identify plausible such attack scenarios, human perceptual abilities play a decisive role. For instance, humans are guided by their senses, making them robust to, e.g., visual and phonetic attacks. Other scenarios to which humans have been shown robust include the removal of vowels from words or the shuffling of characters while keeping the initial and final letters fixed (see Section 2). However, the varieties in which text can be perturbed is certainly far from infinite, as (ordinary) humans, with all their cognitive constraints, still need to be able to decipher the text messages.

In this work, we provide the first large-scale catalogue for *low-level* (orthographic) attack scenarios. Our search is motivated by insights into human cognitive limitations and constraints and encompasses nine different attack modes (some of which are overlapping); cf. Table 1. We then examine the robustness of *RoBERTa* (Liu et al., 2019) to our attacks, finding that its performance can sometimes be severely decreased for our selection of attackers (up to the random guessing baseline); hence we call our benchmark *Zéroe*. The reason may be that our noises are not always natural, in the sense of having high support in large datasets such as CommonCrawl or Wikipedia, but they are still within the limits of cognitive abilities of ordinary humans. Finally, we show that under realistic conditions, standard adversarial training can restore

Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, pages 786–803
December 4 - 7, 2020.

Attacker	Sentence
inner-shuffle	Aadrreavsil aacttks are hmarsels.
full-shuffle	idaAasvrler tstkaac are harmless.
intrude	A d v e r sar ial at:ta:ck:s are h}ar}m}less.
disemvowel	dvrsrl ttcks r hrmlss.
truncate	Adversaria attack are harmles.
segment	Adversarial attacksare harmless.
typo	Adverssrial attaxks are harmless.
natural noise	Adversarial attacs rae harmless.
phonetic	Advorcariel attaks are harmless.
visual	Äḋʊɜ˞rsarīał at̲tacḳṣ āɾe hǟrɰʟẽṣ.

Table 1: Ten different modifications of the sentence "Adversarial attacks are harmless."

RoBERTa's performance only to a limited degree.[1]

2 Related Work

We classify adversarial attacks into *high-* and *low-level* attacks.[2]

Attack Scenarios. There are a variety of works that introduce **low-level** orthographic attacks.[3] Ebrahimi et al. (2017) trick a character-level neural text classification model by flipping the characters which cause most damage. Their approach is white-box, i.e., assumes access to the attack model's parameters. Eger et al. (2019) exchange characters with similar looking ones and show that humans are robust to such visual perturbations, while machines may suffer severe performance drops. Belinkov and Bisk (2017) exchange adjacent letters on the keyboard with each other (keyboard typos) and introduce natural noise based on human typing errors extracted from different Wikipedia edit histories, as well as letter swaps. They use this natural and synthetic noise to show the brittleness of machine translation (MT) systems, which contrasts with corresponding human robustness. Ebrahimi et al. (2018) also fool MT systems with character-level modifications. Tan et al. (2020) attack words by replacing them with morphological variants, which also mostly results in orthographic attacks (in English).

High-level attacks require a deeper understanding of the meaning and the syntactical structure of the sentence. Jin et al. (2019) generate semantically similar and syntactically correct adversarial examples by replacing words with suitable synonyms. Hosseini et al. (2017) and Rodriguez and Rojas-Galeano (2018) attack toxic detection systems by obfuscation, i.e., misspelling of the abusive words (a low-level attack), and via polarization, i.e., inverting the meaning of the sentences by inserting the word "not". Alzantot et al. (2018) introduce an optimization-based algorithm to generate adversarial examples by replacing words in the input. Their generated words are semantically similar because they are nearest neighbors in the GloVe embedding space. They are also syntactically correct because they need to fit into the surrounding context with respect to the 1 billion words language model. Iyyer et al. (2018) generate syntactically correct paraphrases for a sentence. Ribeiro et al. (2018) use MT backtranslation to produce meaning-preserving adversaries. They generate adversarial examples for machine comprehension, sentiment analysis and visual question answering to show robustness issues in state-of-the-art models for each task. Jia and Liang (2017) insert semantically correct but irrelevant paragraphs into texts to fool neural reading comprehension models.

Robustness. Adversarial training is a commonly used technique to address adversarial attacks (Szegedy et al., 2014). The term may refer to calculating model gradients with respect to the input and inserting new training examples based on this gradient (Goodfellow et al., 2014). Alternatively, adversaries obtained from the attacker are inserted at train time (Belinkov and Bisk, 2017; Alzantot et al., 2018; Eger et al., 2019).

3 Catalogue of Attacks

We propose a catalogue of ten different attacks. Our intention is to suggest a maximally inclusive list of potential attacks under the constraint that humans are robust to them.

[1] Code and data are provided at `https://github.com/yannikbenz/zeroe`.

[2] As one reviewer points out, a conceptual difference between high- and low-level attacks is that low-level attacks (as we define them) oftentimes induce linguistically corrupt text which can still be understood by humans, while high-level attacks operate in a noise-free environment to show the brittleness of systems even under 'normal' circumstances.

[3] Low-level adversarial attacks are in part examined by approaches to handle noisy user-generated text (Baldwin et al., 2015), with one difference being that attacks are often malicious in nature and may thus come in different forms.

3.1 Attack protocol

Our attack protocol is *black-box* and *non-targeted* (Xu et al., 2019): we do not assume access to model parameters and our goal is to fool the system without any desired outcome in mind—in contrast, a spammer would want spam emails to be misclassified as non-spam, but not necessarily the reverse.

We parameterize attack levels by a *perturbation probability* $p \in [0, 1]$. With p, our goal is to attack $p \cdot 100\%$ of all tokens in each sample in our dataset. To do so, for each sample $w = (x_1, \ldots, x_n)$, we randomly and without replacement draw a token index i to perturb. We independently flip a coin with tail probability p to determine whether the token x_i should be attacked. We do so until either $p \cdot 100\%$ of all tokens in w are perturbed or else if there are no more indices left.

3.2 Attacks

Some of our attacks, each of which operates on the character-level of an attacked word, are parametrized by a character-level perturbation probability ϕ. For simplicity, we set $\phi = p$ throughout, where p is the above defined word level perturbation probability.

Inner Shuffle. This randomly shuffles all letters in a word except for the first and last. This attacks builds on the human ability to still comprehend words if the first and last letter remain intact (Rayner et al., 2006). We only allow change in words with length ≥ 3.

Full Shuffle. This is the extreme case of the inner-shuffle perturbation where the constraint relating to initial and final letters is dropped. We include this attack for completeness, even though we do not assume high degrees of human robustness to it. We apply this to all words with length ≥ 2.

Intruders. Inserting unobtrusive symbols (Hosseini et al., 2017) in words is a typical phenomenon in social media, e.g., to avoid censorship. Depending on the symbols chosen, an attack may have little effect on humans. We choose the inserted symbol randomly but in case of multiple insertions into one word keep the symbol identical. We allow the following symbols to be inserted: !”#$%&′()*+,−./ :;<=>?@[\]ˆ_‘{|} , including whitespace. The perturbation probability ϕ additionally influences the number of insertions taking place. For each two characters, ϕ indicates how likely the insertion of a symbol between them is. We apply this attack to all words with length ≥ 3.

Disemvoweling. This removes all vowels (a, e, i, o, u) from a word. If a word only consists of vowels, it will be ignored to prevent it from being deleted. Words with length ≤ 3 are skipped to maintain readability. Disemvoweling is a common feature of SMS language and on social media presumed to require little cognitive effort for humans (Boyd et al., 2010).

Truncating. This removes a fixed number of letters from the back of a word. We only cut the last letter from words of length ≥ 3 to maintain readability. Predicting word endings from beginnings is considered an easy task for humans (Elman, 1995).

Segmentation. This joins multiple words together into one word. Here, the perturbation level is the probability to merge the first two adjacent words. Each following word gets a lower probability to get merged (ϕ^2, ..., ϕ^n) to prevent ‘giant’ words. We do not apply this attack to sequence tagging tasks such as POS, because the joined words would have no proper tag, making evaluation more difficult. The ability of humans to segment unsegmented input is already acquired during infancy (Goldwater et al., 2009).

Keyboard Typos. We adopt this attack from Belinkov and Bisk (2017) and adapt it to our workflow. Hereby, adjacent letters on the English keyboard are replaced by each other randomly. This simulates human typing errors. The higher the perturbation probability ϕ, the more characters are exchanged by adjacent letters.

Natural Typos. Words are replaced by natural human errors from the Wikipedia edit history (Belinkov and Bisk, 2017) which contains multiple sources of error: phonetic errors, omissions, morphological errors, key-swap errors and combinations of them.

Phonetic. An ideal phonetic attack leaves the pronunication of a word intact but alters its spelling. Phonetic attacks are common especially in English with its irregular mapping of pronunciation and spelling. They do not only occur as mistakes but also as a form of creative language use (Tagg, 2011).

Visual. Visual attacks are based on the idea that humans may easily recognize similar looking symbols (Eger et al., 2019). We replace each character in the input sequence with one of its 20 visual nearest neighbors in the visual space defined below. This attack is also parameterized by ϕ: we replace each letter in a word i.i.d. randomly with probability ϕ.

We observe that our attacks are *not* directly comparable. For example, at some perturbation level p, truncate removes $O(p \cdot n)$ characters, where n is sentence length. In contrast, intruders inserts $O(p^2 \cdot n \cdot m)$ characters, where m is a bound on word length.

3.3 Implementation of Visual and Phonetic Attacks

We describe details of phonetic and visual attacks below, as they are more involved.

Phonetic Embeddings and Attacks. In order to replace words by phonetically similar ones, we use two stages. First, we train two Seq2Seq models to translate a letter string into its phonetic representation and vice versa. We use the Combilex dataset to do so (Richmond et al., 2010). In addition to that, we induce *phonetic word representations*, i.e., a vector space where two words are close if they are pronounced alike. We use an InferSent-like architecture to do so (Conneau et al., 2017). Details are given in the appendix. When a word x should be phonetically perturbed, we run the first Seq2Seq model to obtain a phonemic representation and then convert this back to a letter string $\tilde{x}$ (as in backtranslation in MT). We finally keep $\tilde{x}$ when it is phonetically similar to x. We added the latter step because we observed that some resulting words $\tilde{x}$ had very different pronunciation than x after the backtranslation.

Visual embeddings. In order to generate visual character embeddings, we used an architecture introduced by Larsen et al. (2016) as a combination of GAN and VAE, called VAEGAN. The model is able to learn embeddings which encode high-level abstract features. This property is desirable in our case, because humans rely on abstract features (Dehaene and Cohen, 2011), i.e., shape and spatial relation of the letter, instead of pixels while reading. The model is described in the appendix.

To obtain visual character embeddings, we generate a grayscale image of size 24×24 for each character in the Basic Multilingual Plane (BMP; 65k characters) of the standard Unicode character set with *Pillow*. The VAEGAN is trained on the full BMP dataset. After training, we compute 256-dimensional visual letter embeddings by encoding the respective letter image with the encoder of the VAEGAN. The quality of the embeddings can be derived via the models' ability to properly reconstruct an image from them, see Figure 7 in the appendix.

4 Experimental Setup

4.1 Base model and datasets

Our base architecture used in all experiments is *RoBERTa* (Liu et al., 2019). RoBERTa is a robustly optimized extension of BERT that has been trained (i) for longer, (ii) on more data, and (iii) without the next sentence prediction task. RoBERTa has been shown to outperform BERT on a variety of benchmark tasks, including those contained in GLUE (Wang et al., 2018). We study the performance of *RoBERTa* in our attack scenarios on three different NLP tasks. Dataset statistics are shown in Table 2.

POS tagging is a sequence tagging task where each token in the input needs to be labeled with its respective POS tag. We use the English universal dependency dataset with 17 different tags (Nivre et al., 2016).

NLI is a classification task in which the relation of a sentence pair must be predicted. Relation labels are *neutral*, *contradiction* and *entailment*. We use SNLI (Bowman et al., 2015).

Toxic Comment Classification (TC) labels sentences (typically from social media platforms) with one or several toxicity classes. Possible labels are: *toxic*, *obscene*, *threat*, *insult* and *identity hate*. For this task, we choose the jigsaw toxic comment challenge dataset from kaggle[4]. The current best performance on the leaderboard has an AUCROC (area under the receiver operations characteristic curve) score of 98.8%.

4.2 Results

We consider the cases of *low* ($p = 0.2$), *mid* ($p = 0.5$) and *high* ($p = 0.8$) attack levels.

In Figure 1, we plot the performance of *RoBERTa* for the three tasks POS, NLI and TC individually as we perturb the test data using our attackers. Detailed numbers are reported in Table 6

[4]https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

Task	Dataset	Train	Test	Clean score
POS Tagging	Universal Dependencies (part)	13k	2k	96.95
NLI	Stanford Natural Language Inference	550k	10k	90.41
Multilabel Classification	Toxic Comment	560k	234k	0.93

Table 2: Overview of the NLP tasks used in this work. Clean scores are scores from training and testing on clean data.

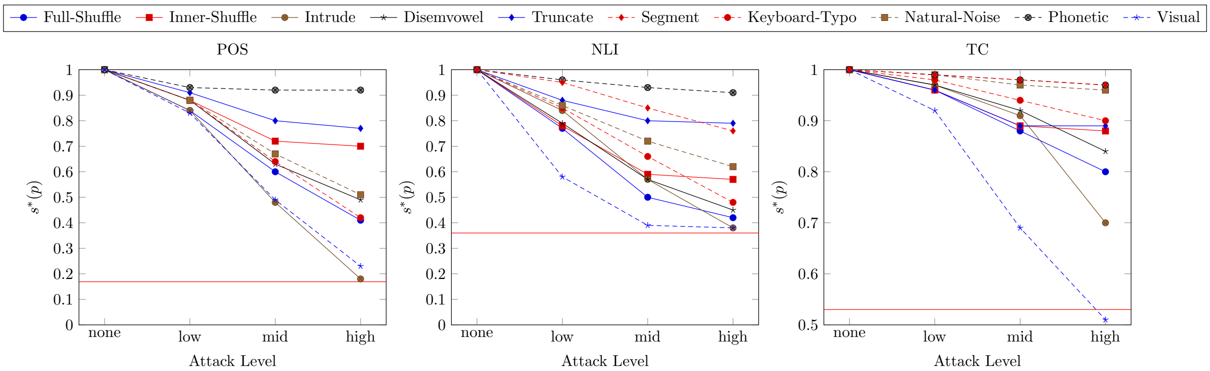

Figure 1: Performance decreases of RoBERTa on the three downstream tasks: POS, SNLI an TC. Red lines indicate the random guessing baseline.

in the appendix. We report scores relative to the model performances on the clean test set:

$$s^*(p) = \frac{s(p)}{s(0)}, \quad p \in \{0, 0.2, 0.5, 0.8\} \qquad (1)$$

where $s(0)$ is the task specific performance on clean data listed in Table 2 and $s(p)$ is the performance for attack level p. Scores are measured in accuracy for POS and NLI, and in AUCROC for TC classification. Clean performance scores depend on the specific task and dataset. For example, NLI has a worst score of around 33% accuracy (majority label) and POS has a corresponding worst score of around 16% accuracy. The worst performance of TC is reached at AUCROC score of 50%—at this point, the model is no longer able to distinguish between the different classes. We mark these values relative to the tasks' best performance ($s(0)$) in Figure 1 as red lines.

Each task suffers performance decreases from each attacker. The higher the perturbation level, the lower the model performance.

The **phonetic** attack is the least effective for all tasks with maximally 10 percentage points (pp) performance decrease with the *high*est perturbation probability of 0.8. The **truncate** attack yields higher performances decreases in all three tasks, being roughly twice as effective. The performance decreases by 10pp from *none* to *low* and additional 10pp from *low* to *mid*. Increasing the attack level beyond that does not cause further harm, especially for NLI and TC. Concerning the **segmentation** attack, for NLI, it leads to a similar performance decrease as the truncate attack for *small* p, but becomes more successful as the perturbation level increases to *mid* and *high*. For TC, the performance decrease is almost identical to the phonetic attack.

We notice a linear decrease in performance for each task when increasing the perturbation level of the **natural-noise** attack. Especially POS and NLI suffer a strong performance deterioration of around 40pp and 50pp for the *high*est attack level. Both lose 15pp to 20pp performance per attack level increase.

Full- and **inner-shuffle** randomize the order in an input word but humans are more robust to inner-shuffle. Full-shuffle also affects *RoBERTa* more than inner-shuffle. It tends to be one of the strongest attack scenarios, while inner-shuffle typically ranks in the midfield.

The **disemvowel** attack has different effects in different tasks. For POS, it is almost identical to the natural-noise attack with a slightly stronger impact of 5pp for *mid* and 3pp for *high* and a maximum on 50pp. NLI loses around 20pp performance on *low* and it decreases an additional 20pp by increasing the level to *mid*, and reaches its greatest decrease by 55pp on *high*. In TC, model performances decrease linearly from *none* to *low* and *mid* by 8pp each. The *high* attack level doubles to a total of 15pp performance loss. The **keyboard-typo** attacks have median impact throughout tasks and attack levels.

The **intrude** attack is among the most severe attacks across all three tasks. For TC, the *low* and *mid* attack levels have a relatively low impact compared to *high* which yields a performance loss of 30pp. It decreases model performance the most on the POS task by above 80pp. Especially for both sentence-based tasks NLI and TC, the **visual** attack decreases are also among the most severe, while *RoBERTa* is marginally more robust on the POS task. Even for the *low* perturbation level, the NLI model suffers from more than 40pp performance decrease. The performance for high p even falls below the red line marked as our lower bound baseline.

4.3 Defenses

In the following, we report the performance increase from shielding the methods with adversarial training:

$$\Delta_\tau(p) := \frac{\sigma(p)}{s(0)} - s^*(p) \qquad (2)$$

where $\sigma(p)$ is the score for each task with one of two defense methods τ:

- **1-1 adversarial training**(α,β): Here, we train on a mixture of *low*, *mid*, *high* attacked data (each perturbation level is roughly equally likely to appear in the training data). We attack with some attacker α and measure performance when the test data is attacked with attacker β.
- **leave-one-out** (LOO): Here, we train on a mix of all attackers except for the one with which the test data is attacked. The train data contains an equal mix of data from each attacker and attack level.

4.3.1 Adversarial Training

1-1 (α, α) In Figure 2, we report the performance of our models each trained on perturbed data and evaluated against *the same kind* of perturbation. This gives an unrealistic upper bound since the defender would have to know how it is being attacked.

For POS, the adversarially trained models lose a bit of their performance on clean data, but their performance on perturbed data improves, especially against intrude and truncate for the *low* attack level. The robustness improvements for the remaining attackers are very similar and range from 3pp increase for the natural-noise attack to 8pp for the disemvowel attack. With one exception, the improvement at large perturbation levels p is highest, and obtains a maximum improvement of 40pp for inner-shuffle.

For NLI, the models again tend perform worse on clean data. As the perturbation level increases, we see a smooth and steady increase of the values $\Delta_\tau(p)$ across all attackers. Improvement is best for intrude which was also among the most damaging attacks.

For TC, model performances increase also on clean data, which is likely due to the nature of the task. As the attack level increases, $\Delta_\tau(p)$ gradually further increases across tasks. For *high*, largest increase is again observed for intrude as well as for visual, which also had largest impact in the non-shielded setting.

1-1 (α, β) In Figure 3, we show all 1-1 values for different combination of attackers on train (α) and test data (β). We see that the diagonal ($\alpha = \beta$) always profits considerably, but the off-diagonal can be positive or negative, depending on the choice of α and β. We clearly see that (1) truncate, disemvowel, keyboard-typo, natural noise, visual, and intruders are similar in the sense that training on them shields against their attacks at test time. (2) Full-shuffle and inner-shuffle form a second group and (3) phonetic attacks a third group. This is to some degree a natural clustering, as (1) removes or replaces characters, (2) destroys the order of words, and (3) modifies entire words using more complex operations. visual is an outlier in group (1), since it improves no matter what attacks are added at train time.

Leave-One-Out Figure 4 shows the performance of our models when trained on a mixture of all attackers except the one evaluated on. This is the most plausible scenario of model defense in the case of an unknown new attack scenario at test time.

For POS, the performance against the phonetic attack remains mostly unchanged, while $\Delta_\tau(p)$ increases as a function of p against natural-noise,

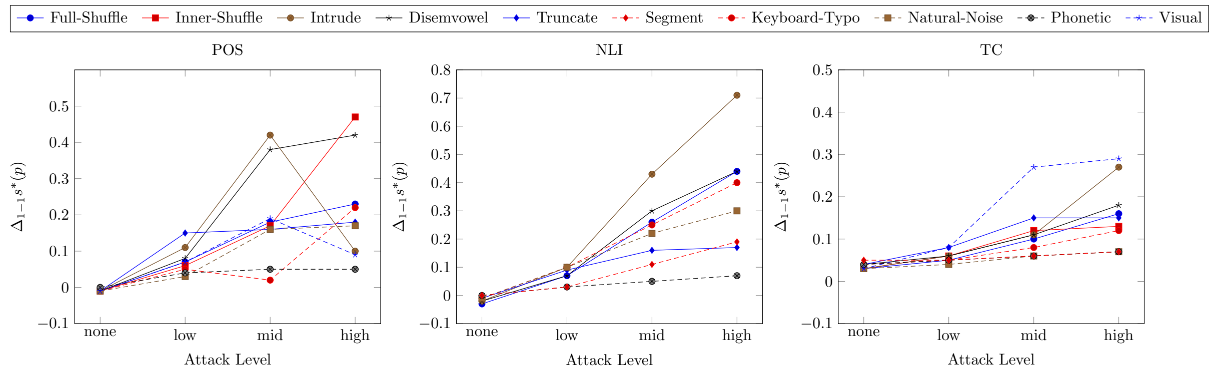

Figure 2: Performance improvements of the models adversarial trained and evaluated individually on the attacker introduced in Section 3 for POS left, NLI mid and TC right. Performance measured in $\Delta_\tau(p)$ defined in Eq. 2.

		FS	IS	INT	DIS	TRUN	KEY	NAT	PH	VIS
FS	low	6.35	-0.46	-2.13	0.49	-0.29	-0.69	-1.21	0.02	0.96
	high	17.59	-0.1	-4.1	4.03	1.81	-0.45	1.48	-1.23	2.24
	mid	21.8	0.48	-4.36	7.59	4.37	0.11	3.93	-1.68	1.97
IS	low	2.01	6.02	0.1	0.72	-0.31	1.13	-0.33	0.14	1.74
	mid	3.7	16.71	0.89	4.24	2.4	3.53	0.41	0.08	4.08
	high	3.77	18.29	1.25	5.25	2.52	3.89	0.59	-0.28	4.78
INT	low	-2.19	-1.04	11.3	0.4	0.63	5.46	0.85	-2.16	5.3
	mid	-7.79	-4.89	41.5	10.77	7.26	12.38	6.71	-2.54	13.93
	high	-7.69	-6.22	62.81	9.8	6.59	11.4	29.56	-2.27	4.3
DIS	low	-2.95	-2.79	-0.56	8.07	0.67	0.19	0.48	-0.97	0.35
	mid	-4.75	-1.42	1.44	27.73	6.05	5.18	5.68	-1.17	3.27
	high	-4.39	0.42	2.76	41.44	9.2	8.68	9.39	-1.82	4.62
TRUN	low	-0.4	-1.07	-0.23	0.4	6.01	0.88	1.46	0.06	0.23
	mid	-0.22	-1.38	0.93	2.18	15.87	3.18	4.93	-0.61	1.58
	high	-0.2	-1.27	1.17	2.79	17.88	3.53	5.73	-0.78	1.72
KEY	low	-1.36	-1.9	0.11	0.3	0.24	5.05	0.91	-0.95	1.93
	mid	-1.89	-1.42	1.4	2.82	1.66	11.04	4.65	-1.79	4.12
	high	-4.07	-2.49	3.96	3.26	4.65	22	6.72	-3.27	5.94
NAT	low	-1.42	-2.77	-0.13	-0.21	-0.6	0.49	3.18	-0.93	1.14
	mid	-0.31	-2.86	0.47	2.77	1.03	2.61	15.62	-0.77	2.15
	high	-2.23	-3.05	2.33	3.46	4.04	4.2	16.7	-1.27	3.19
PH	low	-1.8	-1.96	-1.73	-1.22	-1.41	-1.76	-0.81	4.27	-1.15
	mid	-2.41	-2.18	-1.97	-1.45	-1.73	-2.16	-1.02	5.3	-1.42
	high	-2.21	-2.16	-1.95	-1.44	-1.69	-2.1	-0.95	5.41	-1.39
VIS	low	1.55	1.61	2.58	2.12	2.09	3.69	1.74	0.68	7.43
	mid	5.01	6.65	7.91	9.13	7.41	9.92	5.09	1.18	18.44
	high	1.46	3.62	-0.25	7.87	6.6	7.99	3.81	1.22	8.94

Figure 3: 1-1 (α, β) adversarial training for POS. Column: train, row: test. Numbers give values $\Delta_\tau(p)$, see Eq. (2). Red colors give performance decreases, relative to the results on clean data; blue colors show increases.

inner-shuffle, full-shuffle, truncate and keyboard-typo. The best defense is against natural-noise with 3pp for *low* and 7pp for *mid* and *high*. Shielding against visual, intrude and disemvowel attacks yields lower values $\Delta_\tau(p)$ on attack level *high* compared to *mid*. Overall, we see mild improvements compared to the unshielded situation, but expectedly, these are lower than for 1-1 shielding.

For NLI, the performance against keyboard-typo, full-shuffle, inner-shuffle, natural-noise and truncate exhibits steady improvements with increasing attack level which range from 10pp to 20pp for attack level *mid* and *high*. The performances against intrude and disemvowel also show steady improvements with the attack levels but are generally higher with up to 29pp. For attack level *low*, the performance improvement against the visual attacker is with 20pp more than twice the value of the others. This improvement diminishes in the *mid* and *high* attack levels and even drops there below the improvements against most of the other attackers.

In the TC task, the performance against visual improves even for *low* level to 8pp, increases for *mid* to 23pp and maximizes to 29pp total improvement for attack level *high*. The performance against the intrude attack is also very good: for *low* attack level the improvement (11pp) is even higher compared to visual (8pp). The performances against full-shuffle, inner-shuffle, disemvowel, segment, keyboard-typo, natural-noise and phonetic behave similar for attack level *low* and *mid* with 4pp to 7pp total improvement. Shielding against full-swap and

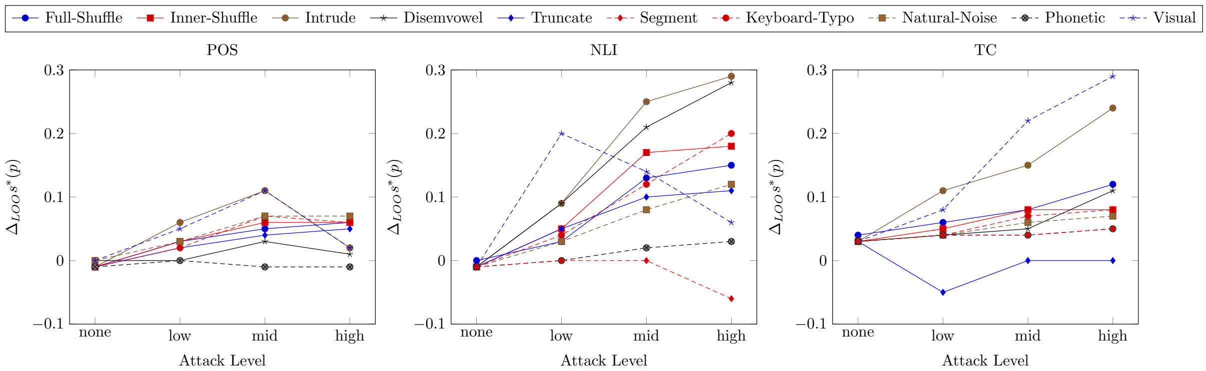

Figure 4: Leave-one-out defense: Performance improvements of the models adversarial trained on all attackers introduced in Section 3 except the one they are evaluated on for POS left, NLI mid and TC right. Performance measured in $\Delta_\tau(p)$ defined in equation 2.

disemvowel is slightly better than the last group. There is no overall positive effect for truncate.

4.4 Discussion

Overall, the phonetic attack was least effective. We assume this is because few words were changed overall as a considerable amount of phonetic replacements were either identical to the input and some were even discarded.

The truncate attack performed better than the phonetic attack in all three tasks but it still remained low overall, possibly as we truncated only by 1 character, leading to small changes in the appearance of a word.

We attribute the low impact of the segmentation attack to *RoBERTa*'s BPE encoding, which apparently allows it to partly de-segment unsegmented input. We observe that some attacks (e.g., segmentation, keyboard-typo, and natural-noise) have less effect in TC compared to POS and NLI, possibly because of higher natural occurrences of these phenomena in the TC dataset.

The intrude and visual attacks are among the strongest. This is *not only* because they are doubly parametrized unlike many others—i.e., for *high* attacks, not only the majority of words is attacked but also the majority of characters within a word—since they are also effective at *low* attack levels. We partly attribute their success to the fact that they cause a high out-of-vocabulary rate for *RoBERTa* and tend to increase the number of input tokens, as they cause *RoBERTa* to segment the input at unknown characters. This may lead to the number of input tokens exceeding *RoBERTa*'s builti-in max token size, leading to cutting off the ending of the sentence.

Rank	**POS**		**NLI**		**TC**	
1	1-1 (α, α)	16	1-1	20	1-1	12
2	LOO	4	LOO	10	LOO	9
3	1-1 (α,β)	1	1-1	3	1-1	7

Table 3: Different defense approaches ranked by the average robustness improvement over all attackers. Improvement in percentage points (pp; rounded).

In Table 5 (appendix), attacks are ranked (for *high* attack level) by the performance degradation caused to the model for each individual task. In line with our previous discussion, the visual and the intrude attackers are always the both best performing, followed by full-shuffle (which we deemed as unrealistic as it would also destroy human perception abilities). Figure 5 shows the relationship between the amount of text perturbed in a test dataset and the performance deterioration a model suffers. This shows a clear (linear) trend and indicates that a successful attacker most importantly needs to attack many characters of a text to be effective, despite all individual qualitative differences between the attackers discussed above.

In Table 3, a ranking of defense strategies is given. 1-1 (α, α) performs best, but is unrealistic. LOO is a robust alternative for unknown new attacks. The effectiveness of LOO as defense is also a further justification for designing multiple attack models.

5 Conclusion

We provided the first large-scale catalogue for low-level adversarial attacks, providing a new simple benchmark for testing real-world robustness of future deep learning models. We further showed

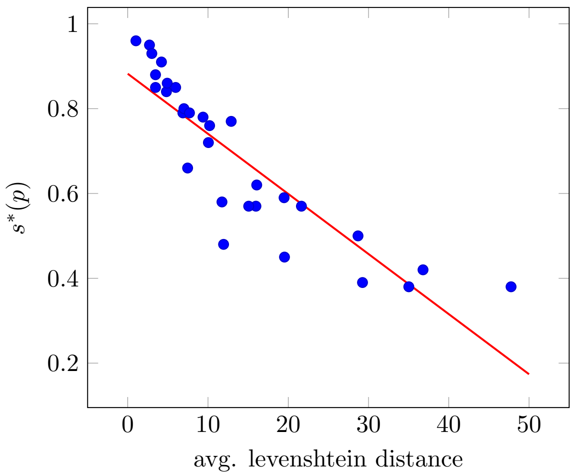

Figure 5: Relation between the amount of text perturbed (measured in edit distance) in a test data set and $s^*(p)$, the performance decrease a model suffers.

that one of the currently most successful deep learning paradigms, *RoBERTa*, is not robust to our benchmark, sometimes suffering catastrophic failure. While many of our errors could probably be addressed by placing a correction layer in front of *RoBERTa* (Choudhury et al., 2007; Pruthi et al., 2019), we believe that our findings shed further light on the differences between human and machine text processing, which deep models eventually will have to *innately* overcome for true AI to become a viable prospect.

Acknowledgments

We thank the anonymous reviewers for their useful comments and suggestions. Steffen Eger has been funded by the HMWK (Hessisches Ministerium für Wissenschaft und Kunst) as part of structural location promotion for TU Darmstadt in the context of the Hessian excellence cluster initiative "Content Analytics for the Social Good" (CA-SG).

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating Natural Language Adversarial Examples. *arXiv e-prints*, page arXiv:1804.07998.

Timothy Baldwin, Marie-Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. 2015. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 126–135.

Yonatan Belinkov and Yonatan Bisk. 2017. Synthetic and Natural Noise Both Break Neural Machine Translation. *arXiv preprint arXiv:1711.02173*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Danah Boyd, Scott Golder, and Gilad Lotan. 2010. Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. In *Proceedings of the 2010 43rd Hawaii International Conference on System Sciences*, HICSS '10, page 1–10, USA. IEEE Computer Society.

Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *International Journal of Document Analysis and Recognition (IJDAR)*, 10(3-4):157–174.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. *arXiv e-prints*, page arXiv:1705.02364.

Stanislas Dehaene and Laurent Cohen. 2011. The unique role of the visual word form area in reading. *Trends in Cognitive Sciences*, 15(6):254 – 262.

Javid Ebrahimi, Daniel Lowd, and Dejing Dou. 2018. On Adversarial Examples for Character-Level Neural Machine Translation. *arXiv e-prints*, page arXiv:1806.09030.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2017. HotFlip: White-Box Adversarial Examples for Text Classification. *arXiv e-prints*, page arXiv:1712.06751.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding NLP systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647, Minneapolis, Minnesota. Association for Computational Linguistics.

Jeffrey L Elman. 1995. Language as a dynamical system. *Mind as motion: Explorations in the dynamics of cognition*, pages 195–223.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2009. A bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21 – 54.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and Harnessing Adversarial Examples. *arXiv e-prints*, page arXiv:1412.6572.

Hossein Hosseini, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. 2017. Deceiving Google's Perspective API Built for Detecting Toxic Comments. *arXiv e-prints*, page arXiv:1702.08138.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. *arXiv e-prints*, page arXiv:1804.06059.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 2021–2031.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *arXiv e-prints*, page arXiv:1907.11932.

Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. 2016. Autoencoding beyond pixels using a learned similarity metric. In *33rd International Conference on Machine Learning, ICML 2016*, volume 4, pages 2341–2349.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv e-prints*.

Joakim Nivre, Marie Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation, LREC 2016*, pages 1659–1666.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5582–5591, Florence, Italy. Association for Computational Linguistics.

Keith Rayner, Sarah J. White, Rebecca L. Johnson, and Simon P. Liversedge. 2006. Raeding wrods with jubmled lettres: There is a cost. *Psychological Science*, 17(3):192–193.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.

Korin Richmond, Robert Clark, and Sue Fitt. 2010. On generating combilex pronunciations via morphological analysis. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association, INTERSPEECH 2010*, pages 1974–1977.

Nestor Rodriguez and Sergio Rojas-Galeano. 2018. Shielding Google's language toxicity model against adversarial attacks. *arXiv e-prints*, page arXiv:1801.01828.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*.

Caroline Tagg. 2011. Wot did he say 01" could u not c him 4 dust?: Written and spoken creativity in text messaging. *Transforming literacies and language: Multimodality and literacy in the new media age*, page 223.

Samson Tan, Shafiq Joty, Min-Yen Kan, and Richard Socher. 2020. It's morphin' time! Combating linguistic discrimination with inflectional perturbations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2920–2935, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Han Xu, Yao Ma, Haochen Liu, Debayan Deb, Hui Liu, Jiliang Tang, and Anil K. Jain. 2019. Adversarial Attacks and Defenses in Images, Graphs and Text: A Review. *arXiv e-prints*, page arXiv:1909.08072.

Homophone	byte	bite
Abbreviation	I love you too!	I luv U 2!

Table 4: Example of a homophone and a typical "internet slang" abbreviation.

A Appendices

A.1 Phonetic and visual embeddings

Phonetic Word Embeddings. To induce phonetic word embeddings, we adopt the Siamese network of InferSent (Conneau et al., 2017). InferSent was originally designed to induce vector representations for two sentences from which their entailment relation was inferred. We adapt InferSent to encode two words so that their phonological similarity can be inferred: *identical*, *very similar*, *similar* and *different*. We use the BiLSTM max-pooling approach from the original InferSent paper, where we set the induced phonetic embeddings size to 100.

We build our own dataset for phonetic similarity by leveraging data from different sources. Initially, we use Combilex (Richmond et al., 2010), which gives phonetic representations for standard (American) English words. We calculate the normalized edit distance between the phonemes of each word pair to determine the phonetic similarity of two words:

$$\text{sim}_{\text{ph}}(\pi_1, \pi_2) = 1 - \frac{d(\pi_1, \pi_2)}{\min(|\pi_1|, |\pi_2|)} \quad (3)$$

where π_i are phonetic sequences for underlying words and d is the edit-distance. We then map the words into 4 different classes: *identical* ($\text{sim}_{\text{ph}} = 0$), *very similar* ($0 < \text{sim}_{\text{ph}} < 0.1$), *similar* ($0.1 < \text{sim}_{\text{ph}} < 0.3$) and *different* ($0.3 < \text{sim}_{\text{ph}}$). To keep the training data for each class more balanced, we added handcrafted and crawled samples, e.g., homophones. We also wanted to include "internet slang" style phonetic replacements like in Table 4. We therefore crawled them and added them to the bins *identical* and *very similar* based upon manual inspection. Overall, we compiled 5k examples for each of our four labels. The *similar* and *different* bins consist only of data from Combilex, whereas the *identical* and *very similar* bin contains 1.3k samples from Combilex and 3.7k crawled samples. References for crawled sites are given in A.2.

Visual Embeddings. The model reduces the dimension of input x, e.g., an image, by applying multiple convolutional steps in the *encoder* to compute the latent representation z of x. Afterwards, it reconstructs the original input x in the *decoder* by applying multiple deconvolutional steps to z. This reconstructed version of x is called $\tilde{x}$. Additionally, a second input z_p sampled from $\mathcal{N}(0, I)$ is inserted into the *generator* to obtain x_p. *Decoder* and *generator* perform the same task on different inputs; they can be considered as identical and therefore share their parameters. The *discriminator* takes $x, \tilde{x}$ and x_p as inputs and discriminates which input is a real training sample and which is a fake. Figure 6 illustrates the working of the architecture.

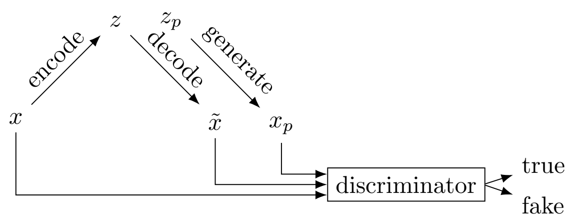

Figure 6: Schematic representation of Variational Autoencoder Generative Adversarial Network (VAEGAN) taken and adapted from Larsen et al. (2016). z can be decomposed as $z = \mu + \sigma$ and is used to sample $z_p = \mu + \sigma\epsilon$ where ϵ is noise defined as $\epsilon \sim \mathcal{N}(0, I)$

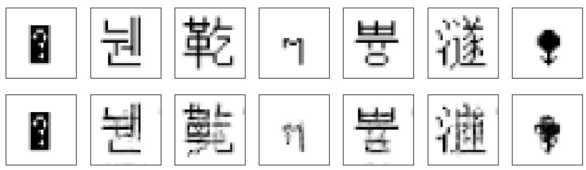

Figure 7: Reconstruction of images after being compressed to its latent representation and decompressed back to the original data distribution.

Figure 8 gives an impression of the encoded visual similarity.

A.2 Homophone resources

List of used resources to gather homophones.

- https://7esl.com/homonyms/
- https://www.englishclub.com/pronunciation/homophones-list.html
- https://www.thoughtco.com/homonyms-homophones-and-homographs-a-b-1692660
- http://www.singularis.ltd.uk/bifroest/misc/homophones-list.html

Ranking	POS	NLI	TC
1	Intrude	Visual & Intrude	Visual
2	Visual	-	Intrude
3	Full-Shuffle	Full-Shuffle	Full-Shuffle
4	Keyboard-Typo	Disemvowel	Disemvowel
5	Disemvowel	Keyboard-Typo	Inner-Shuffle
6	Natural-Noise	Inner-Shuffle	Truncate
7	Inner-Shuffle	Natural-Noise	Keyboard-Typo
8	Truncate	Segment	Natural-Noise
9	Phonetic	Truncate	Segment
10	-	Phonetic	Phonetic

Table 5: Ranking on harmfulness of the attackers on POS, NLI, TC on attack level *high*.

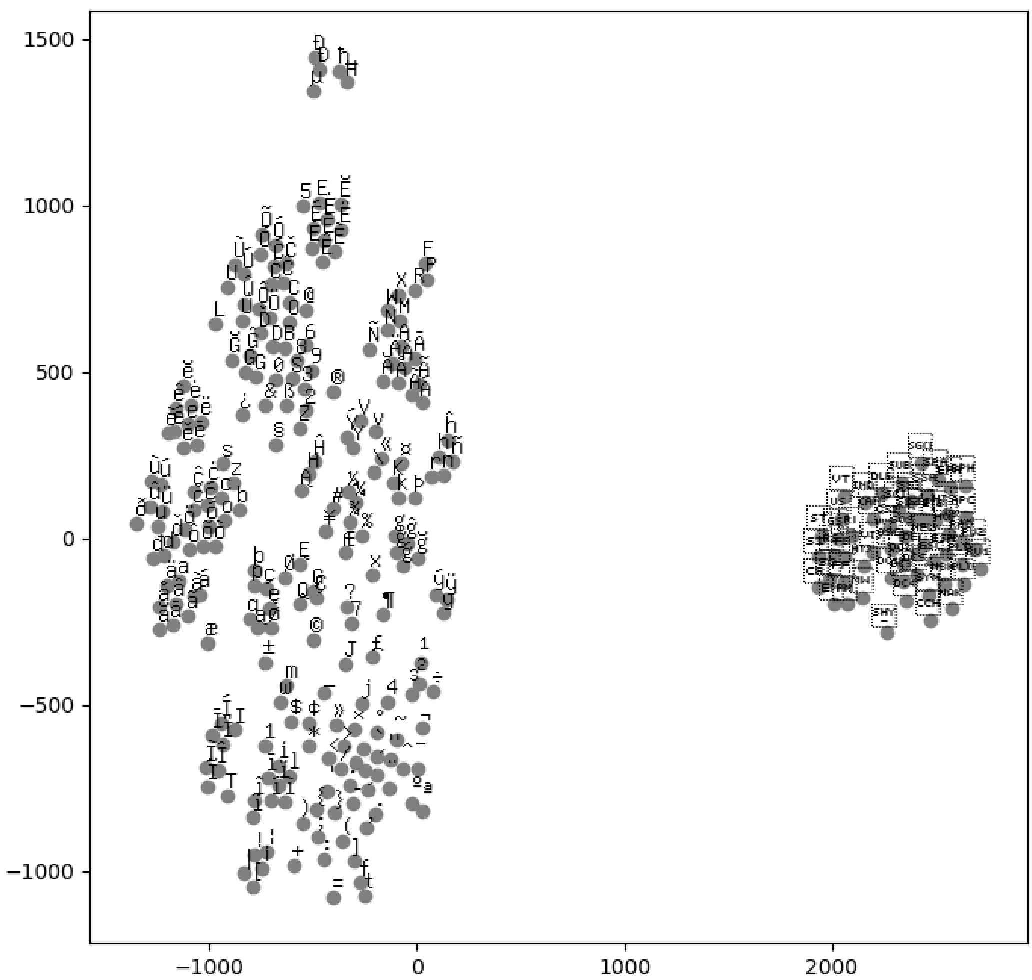

Figure 8: tSNE plot of our character embedding space. As can be seen similar looking characters are clustered.

- https://web.archive.org/web/20160825095711/
- http://people.sc.fsu.edu/~jburkardt/fun/wordplay/multinyms.html
- http://homophonelist.com/homophones-list/
- https://web.archive.org/web/20160825095711/
- http://homophonelist.com/homophones-list/
- https://www.webopedia.com/quick_ref/textmessageabbreviations.asp
- https://www.smart-words.org/abbreviations/text.html
- https://en.wiktionary.org/wiki/Appendix:English_dialect-independent_homophones
- https://en.wiktionary.org/wiki/Appendix:English_dialect-dependent_homophones

A.3 Detailed Result Tables

Hyperparameters of our models can be found in the github accompanying the publication (https://github.com/yannikbenz/zeroe). The following tables give detailed results of our experiments.

Attack	Mode	Accuracy		AUCROC
		POS	NLI	TC
None	-	96.65	90.41	0.93
Full-Swap	low	82.14	70.35	0.90
	mid	58.14	45.70	0.83
	high	40.47	38.35	0.74
Inner-Swap	low	85.96	67.70	0.90
	mid	70.53	53.35	0.83
	high	67.95	51.55	0.82
Intrude	low	81.42	75.97	0.91
	mid	46.91	52.25	0.85
	high	18.15	34.70	0.66
Disemvowel	low	85.24	72.24	0.91
	mid	61.50	51.62	0.86
	high	44.69	41.00	0.79
Truncate	low	88.57	79.83	0.90
	mid	77,40	72.87	0.84
	high	75,11	72.02	0.83
Segment	low	-	86.08	0.93
	mid	-	77.53	0.92
	high	-	69.14	0.91
Keyboard-Typo	low	85.06	76.93	0.92
	mid	62.41	60.21	0.88
	high	40.99	44.16	0.84
Natural Noise	low	85.34	78.43	0.92
	mid	65.36	65.60	0.91
	high	50.06	56.31	0.90
Phonetic	low	90.62	87.40	0.93
	mid	89.09	84.75	0.92
	high	88.95	82.80	0.91
Visual	low	80.52	53.07	0.86
	mid	48.14	35.26	0.64
	high	22.44	34.37	0.48

Table 6: Attacks against unshielded model.

Train / Test	level	FS	IS	INT	DIS	TRUN	SEG	KEY	NAT	PH	VIS	POS	NLI	TC
FS	none	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	95.57	89.56	0.97
	low											84.49	73.05	0.95
	mid											63.48	57.54	0.90
	high											45.72	51.73	0.86
IS	none	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	95.66	88.94	0.96
	low											88.29	75.51	0.94
	mid											75.90	69.07	0.91
	high											73.68	68.54	0.90
INT	none	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	95.65	88.90	0.96
	low											87.54	84.27	0.95
	mid											57.58	74.92	0.93
	high											19.44	61.07	0.84
DIS	none	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	95.69	89.42	0.96
	low											86.00	80.00	0.94
	mid											64.39	70.98	0.91
	high											48.65	66.60	0.89
TRUN	none	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	95.49	89.17	0.96
	low											89.98	84.55	0.84
	mid											81.23	81.97	0.83
	high											79.38	81.62	0.82
SEG	none	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	-	89.02	0.96
	low											-	85.38	0.96
	mid											-	76.92	0.95
	high											-	62.83	0.95
KEY	none	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	95.61	88.80	0.96
	low											87.71	80.47	0.95
	mid											68.64	70.69	0.94
	high											46.51	61.83	0.92
NAT	none	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	95.72	88.67	0.96
	low											88.17	81.27	0.96
	mid											72.30	73.05	0.96
	high											56.78	67.40	0.96
PH	none	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	95.30	88.95	0.96
	low											89.74	87.54	0.96
	mid											87.82	86.27	0.95
	high											87.72	85.34	0.95
VIS	none	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	95.72	89.02	0.96
	low											85.18	70.77	0.93
	mid											58.94	48.80	0.85
	high											24.99	40.22	0.75

Table 7: Adversarial training: leave-one-out.

Test \ Train	level	FS	IS	INT	DIS	TRUN	KEY	NAT	PH	VIS
Clean	-	95.17	95.18	95.04	95.44	95.48	95.40	95.40	96.06	95.66
FS	low	88.49	81.68	80.01	82.63	81.85	81.45	80.93	82.16	83.10
	mid	75.73	58.04	54.04	62.17	59.95	57.69	59.62	56.91	60.38
	high	62.27	40.95	36.11	48.06	44.84	40.58	44.40	38.79	42.44
IS	low	87.97	91.98	86.06	86.68	85.65	87.09	85.63	86.10	87.70
	mid	74.23	87.24	71.42	74.77	72.93	74.06	70.94	70.61	74.61
	high	71.72	86.24	69.20	73.20	70.47	71.84	68.54	67.67	72.73
INT	low	79.23	80.38	92.72	81.82	82.05	86.88	82.27	79.26	86.72
	mid	39.12	42.02	88.41	57.68	54.17	59.29	53.62	44.37	60.84
	high	10.46	11.93	80.96	27.95	24.74	29.55	47.71	15.88	22.45
DIS	low	82.29	82.45	84.68	93.31	85.91	85.43	85.72	84.27	85.59
	mid	56.75	60.08	62.94	89.23	67.55	66.68	67.18	60.33	64.77
	high	40.30	45.11	47.45	86.13	53.89	53.37	54.08	42.87	49.31
TRUN	low	88.17	87.50	88.34	88.97	94.58	89.45	90.03	88.63	88.80
	mid	77.18	76.02	78.33	79.58	93.27	80.58	82.33	76.79	78.98
	high	74.91	73.84	76.28	77.90	92.99	78.64	80.84	74.33	76.83
KEY	low	83.70	83.16	85.17	85.36	85.30	90.11	85.97	84.11	86.99
	mid	60.52	60.99	63.81	65.23	64.07	73.45	67.06	60.62	66.53
	high	36.92	38.50	44.95	44.25	45.64	62.99	47.71	37.72	46.93
NAT	low	83.92	82.57	85.21	85.13	84.74	85.83	88.52	84.41	86.48
	mid	65.05	62.50	65.83	68.13	66.39	67.97	80.98	64.59	67.51
	high	47.83	47.01	52.39	53.52	54.10	54.26	66.76	48.79	53.25
PH	low	88.82	88.66	88.89	89.40	89.21	88.86	89.81	94.89	89.47
	mid	86.68	86.91	87.12	87.64	87.36	86.93	88.07	94.39	87.67
	high	86.74	86.79	87.00	87.51	87.26	86.85	88.00	94.36	87.56
VIS	low	82.07	82.13	83.10	82.64	82.61	84.21	82.26	81.20	87.95
	mid	53.15	54.79	56.05	57.27	55.55	58.06	53.23	49.32	66.58
	high	23.90	26.06	22.19	30.31	29.04	30.43	26.25	23.66	31.38

Table 8: Part-of-Speech tagging adversarial training: 1-1.

Test \ Train	level	FS	IS	INT	DIS	TRUN	SEG	KEY	NAT	PH	VIS
Clean	-	87.54	-	88.29	88.59	88.91	89.90	89.17	89.12	90.24	-
FS	low	83.04	-	64.65	63.28	60.38	62.67	69.46	68.46	66.87	-
	mid	78.58	-	47.62	48.15	42.21	46.05	52.63	50.81	48.11	-
	high	76.96	-	42.75	44.52	39.16	41.55	47.77	46.21	41.53	-
IS	low	81.31	-	71.32	66.40	59.81	63.26	72.23	72.25	66.40	-
	mid	78.82	-	64.82	58.42	51.17	53.63	63.21	64.24	57.16	-
	high	78.08	-	64.06	58.25	50.86	53.53	62.72	62.89	56.61	-
INT	low	76.22	-	85.83	72.49	72.97	72.78	83.73	80.83	74.83	-
	mid	58.99	-	82.61	53.87	51.09	48.45	69.53	62.84	51.53	-
	high	43.22	-	80.76	39.93	36.18	36.60	48.77	40.91	37.07	-
DIS	low	79.14	-	76.27	86.56	67.51	72.52	78.96	77.77	72.65	-
	mid	72.47	-	67.43	84.70	57.60	56.88	69.72	64.85	57.03	-
	high	69.45	-	63.90	84.16	54.50	48.25	65.29	58.44	48.92	-
TRUN	low	81.63	-	84.31	79.66	88.15	80.02	86.35	84.79	80.46	-
	mid	77.87	-	82.45	75.86	87.52	76.11	84.08	81.83	76.19	-
	high	77.25	-	82.46	75.10	87.44	75.79	83.80	81.76	75.80	-
SEG	low	82.32	-	84.32	83.75	84.00	89.07	86.15	85.53	85.69	-
	mid	68.85	-	76.41	75.84	77.33	87.54	80.28	79.37	78.14	-
	high	50.94	-	64.98	68.11	71.36	86.42	73.39	73.19	71.88	-
KEY	low	74.23	-	76.90	71.38	69.95	73.10	86.63	81.04	74.46	-
	mid	57.37	-	62.86	55.62	54.30	58.67	82.98	70.74	56.98	-
	high	45.87	-	52.26	46.29	44.75	47.07	79.82	61.76	44.54	-
NAT	low	77.87	-	78.32	73.62	73.50	75.51	82.39	87.67	76.47	-
	mid	67.98	-	67.98	62.27	60.10	62.97	74.25	85.45	62.85	-
	high	59.73	-	60.81	55.16	53.18	55.41	69.52	84.06	54.89	-
PH	low	85.68	-	85.98	84.23	85.36	86.53	87.50	87.80	89.93	-
	mid	84.25	-	84.21	80.98	82.67	84.17	85.98	86.50	89.40	-
	high	83.07	-	82.71	80.28	81.68	82.68	84.74	85.42	89.19	-
VIS	low	59.79	-	72.33	55.74	56.09	56.91	70.65	66.32	55.34	-
	mid	41.82	-	50.95	37.87	37.01	36.38	45.21	41.84	36.31	-
	high	37.42	-	39.08	33.81	34.26	34.51	36.04	35.37	34.10	-

Table 9: Natural language inference adversarial training: 1-1.

Test \ Train	level	FS	IS	INT	DIS	TRUN	SEG	KEY	NAT	PH	VIS
Clean	-	0.96	0.96	0.96	0.97	0.97	0.98	0.97	0.96	0.97	0.96
FS	low	0.94	0.94	0.94	0.94	0.94	0.95	0.95	0.93	0.95	0.93
	mid	0.92	0.90	0.87	0.88	0.87	0.87	0.89	0.87	0.87	0.88
	high	0.90	0.86	0.80	0.81	0.79	0.79	0.83	0.80	0.77	0.83
IS	low	0.94	0.95	0.94	0.94	0.94	0.95	0.94	0.93	0.94	0.93
	mid	0.92	0.94	0.89	0.90	0.89	0.89	0.91	0.88	0.89	0.89
	high	0.91	0.94	0.88	0.90	0.88	0.88	0.90	0.87	0.88	0.89
INT	low	0.95	0.95	0.96	0.96	0.96	0.96	0.96	0.95	0.96	0.95
	mid	0.92	0.92	0.95	0.92	0.92	0.91	0.94	0.90	0.91	0.91
	high	0.81	0.80	0.91	0.80	0.76	0.75	0.81	0.76	0.72	0.83
DIS	low	0.94	0.95	0.95	0.96	0.94	0.96	0.95	0.94	0.95	0.94
	mid	0.91	0.91	0.91	0.96	0.90	0.90	0.92	0.90	0.89	0.90
	high	0.88	0.88	0.88	0.95	0.86	0.83	0.89	0.86	0.83	0.88
TRUN	low	0.95	0.95	0.96	0.96	0.97	0.96	0.97	0.94	0.96	0.95
	mid	0.94	0.94	0.95	0.94	0.97	0.95	0.96	0.93	0.95	0.94
	high	0.94	0.94	0.95	0.94	0.97	0.95	0.96	0.93	0.95	0.94
SEG	low	0.96	0.96	0.96	0.96	0.97	0.97	0.97	0.96	0.97	0.95
	mid	0.95	0.95	0.95	0.96	0.96	0.97	0.96	0.95	0.96	0.94
	high	0.94	0.94	0.94	0.95	0.95	0.97	0.95	0.93	0.95	0.93
KEY	low	0.95	0.95	0.95	0.95	0.96	0.96	0.96	0.95	0.96	0.95
	mid	0.92	0.92	0.93	0.93	0.93	0.94	0.95	0.92	0.93	0.92
	high	0.88	0.88	0.91	0.89	0.90	0.89	0.95	0.89	0.87	0.90
NAT	low	0.96	0.96	0.96	0.96	0.97	0.97	0.97	0.96	0.97	0.95
	mid	0.95	0.95	0.96	0.96	0.96	0.97	0.97	0.96	0.96	0.95
	high	0.95	0.95	0.95	0.95	0.96	0.96	0.96	0.96	0.96	0.94
PH	low	0.96	0.96	0.96	0.96	0.97	0.97	0.97	0.95	0.97	0.95
	mid	0.95	0.95	0.95	0.96	0.96	0.97	0.96	0.95	0.97	0.95
	high	0.95	0.95	0.95	0.95	0.96	0.96	0.96	0.94	0.97	0.94
VIS	low	0.92	0.93	0.94	0.93	0.93	0.93	0.94	0.91	0.92	0.93
	mid	0.82	0.81	0.86	0.80	0.78	0.77	0.83	0.76	0.71	0.90
	high	0.70	0.69	0.75	0.65	0.62	0.62	0.66	0.64	0.55	0.85

Table 10: Toxic comment adversarial training: 1-1.