UNIVERSITÄT DES SAARLANDES
Prof. Dr. Dietrich Klakow
Lehrstuhl für Signalverarbeitung
NNIA Winter Term 2019/2020

Exercise Sheet 7

Philip Georgis [s8phgeor], Pauline Sander [s8pasand], Vilém Zouhar [vizo00001]

(Solutions)

Deadline: 12. 1. 2021

Exercises

Exercise 7.1 - Norm Penalty

a) Assuming the output of the layer is some distribution, biases shift the distribution, but do not change its shape (they only influence one variable). It is also easier to approximate the bias better (less data is needed), simply by shifting the distribution to true distribution mean. We need to regularize the weights, because they model the interaction between variables, which can pick up noise from the training data, increase training and decrease test performance (overfitting). Biases, on the other hand, are less prone to overfitting. Regularizing them could lead to underfitting.

b) The *Class* coefficient would be penalized more, because without regularization it is much higher (17.28) than for the *Age* variable (0.03). The new loss function would be a sum of upward facing parabolas. This is the reason why input variables are sometimes z-normed. The larger variable would be penalized more in an absolute value, but less proportionally to the original value, because it is more important. This can be demonstrated with the original code. The new coefficients are: 0.03 and -17.24, which are 97% and 99% of the original values.

c) L^1 regularization can lead to results with sparse parameters. This can be seen in Fig. 1, where the L^1 norm leads to w_1 being 0. Since their parameters are zero, we can remove the more irrelevant features altogether. We select the more 'important' features.

Exercise 7.2 - Dataset Augmentation

a) For vision tasks, it is relatively easy (methodically and computationally) to create augmented data with affine transformations or any other filter. The NLP data augmentation methods require more care, time and sometimes even whole pipeline. An example of this would be backtranslation. If this would be done on the go, then it would mean loading the reverse model into GPU, translating and decoding it and finally filtering by quality. It makes much more sense to do this beforehand with large batches.

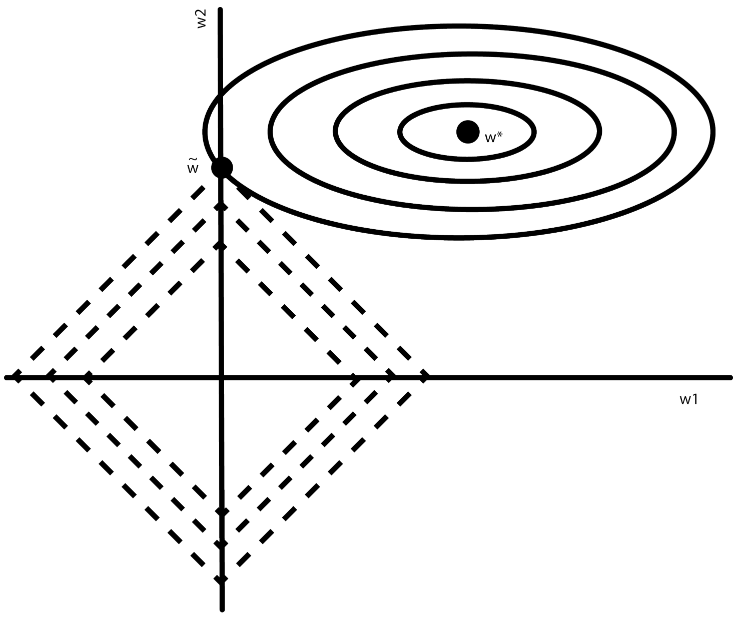

Figure 1: L^1 regularization

b) For machine translation and similar tasks, such as MT quality estiation, the backtranslation is a double-edged sword. Sometimes the corpus (e.g. ParaCrawl) may already be a product of publicly available MT systems. Translating it yet again leads to translationese, which hurts the language modelling capabilities of the model. On the other hand, model distillation has been shown to provide significant boosts, especially in early epochs of training.

c) If we perform it before splitting the data, we have to also keep the original data around, because they are used for evaluation and we want to evaluate on authentic data. Performing it after solves this issue, but we will end up augmenting the same example $k - 1$ times.

d) The size of the training set does not *necessarily* determine the proportion of the data to be augmented. However, given that models are more prone to overfitting on smaller training sets, higher proportions of augmented data would certainly be beneficial in such situations by significantly increasing the amount of available training data. Larger datasets do also benefit from augmentation, though augmenting every item in a large dataset would not be as beneficial against overfitting as doing the same on a smaller dataset.

Exercise 7.3 - Bagging and Dropout

a) In dropout we obscure (randomly) some of the weights for a particular example, which can be seen as training multiple ensembled networks with unobscure weights shared. This is a bit similar to bagging, where each network is trained on sampled dataset in which case there is a probability that there are elements it will never encounter.

b) No, because we want to make use of all of the features given to us. For vanilla dropout we need to multiply the whole layer by the probability so that the output distribution shape is the same.

c)
```
def inverse_dropout(layer, prob):
    # matrix of [0, 1]
    mask = rand(shape=layer.shape) < prob
    return ( layer @ mask ) / ( 1 - prob )
```

Exercise 7.4 - Adversarial Training in NLP

In computer vision, small amounts of noise added to images can result in erroneous outputs (e.g. misclassification or incorrect identification of images), despite being visually imperceptible to humans. In the NLP domain, even small perturbations are not completely imperceptible to humans. Methods for adversarial training in NLP, thus, parallel those of data augmentation in order for small changes not to disrupt the predictions of a model.

a) **Synonym Replacement:** This technique substitutes individual words in a text with semantically related words. The result is a new training example with virtually unchanged meaning, simply using different words, which can increase the robustness of the model, e.g.:
Original: *This **film** displayed a **wealth** of inspiration on the part of the director.*
Adversarial: *This **movie** displayed an **abundance** of inspiration on the part of the director.*

b) **Character Changes:** The effect of adding random noise to images can be modelled by intentionally misspelling words or changing individual characters in a text. This is performed in such a way that it models common misspellings or typos, so that it causes no comprehension problems for human readers, yet would trip up the model, e.g.:
Original: *Their **role** in the process should **definitely** not be discounted.*
Adversarial: *Their **roll** in the process should **definietly** not be discounted.*

c) **Distractor Sentences:** Adversarial examples can also be generated by inserting distracting sentences, which would mislead a model but not a human reader. For example, given a text "*Following her studies in Athens, Maria was soon offered a job in Berlin. Maria moved to Berlin in 2007.*" and the question: "*Where did Maria move in 2007?*", an adversarial example can be constructed by adding to the original text a distractor sentence, which perturbs the sentence which contains the correct answer, e.g. "*Katerina moved to Thessaloniki in 2007.*". This could cause a QA model to answer "Thessaloniki" rather than "Berlin".

d) **Syntactic Perturbations:** This technique creates adversarial training examples by perturbing the syntactic structure of sentences without changing their meaning. For example, hedges or fillers might be inserted, or constituents might be reordered to produce a semantically equivalent but distorted sentence, e.g.:
Original: *The best student in the class is eating lunch by himself.*
Adversarial: *You know, the student eating lunch by himself is perhaps the best in the class.*

Sources:
https://nlpblog.cl.uni-heidelberg.de/index.php/2019/09/20/adversarial-training/
https://towardsdatascience.com/what-are-adversarial-examples-in-nlp-f928c574478e
https://nlp.stanford.edu/pubs/jia2017adversarial.pdf