SimLoss: Class Similarities in Cross Entropy

Konstantin Kobs, Michael Steininger, Albin Zehe, Florian Lautenschlager, and Andreas Hotho

Julius-Maximilians University Würzburg
{kobs,steininger,zehe,lautenschlager,hotho}@informatik.uni-wuerzburg.de

Abstract. One common loss function in neural network classification tasks is Categorical Cross Entropy (CCE), which punishes all misclassifications equally. However, classes often have an inherent structure. For instance, classifying an image of a rose as "violet" is better than as "truck". We introduce SimLoss, a drop-in replacement for CCE that incorporates class similarities along with two techniques to construct such matrices from task-specific knowledge. We test SimLoss on Age Estimation and Image Classification and find that it brings significant improvements over CCE on several metrics. SimLoss therefore allows for explicit modeling of background knowledge by simply exchanging the loss function, while keeping the neural network architecture the same.[1]

Keywords: Cross Entropy · Class Similarity · Loss Function.

Roses are red, violets are blue,
both are somehow similar, but the classifier has no clue.
(Common proverb)

1 Introduction

One common loss function in neural network classifiers is Categorical Cross Entropy (CCE). CCE tries to maximize the assigned target class probability and punishes every misclassification in the same way, independent of other information about the predicted class. Often, however, classes have a special order or are similar to each other, such as different flowers in image classification. Including class similarities using the inherent class structure (e.g., class order), class properties (e.g., class names) or external information about the classes (e.g., knowledge graphs) in the training procedure would allow the classifier to make less severe mistakes as it learns to predict similar classes.

In this work, we modify Categorical Cross Entropy and propose Similarity Based Loss (SimLoss) as a way to explicitly introduce background knowledge into the training process, as visualized in Figure 1. For this, we augment CCE with a matrix containing class similarities and propose two techniques in order to prepare such matrices that exploit certain class relations: class order and general

[1] Code and additional resources: https://github.com/konstantinkobs/SimLoss.

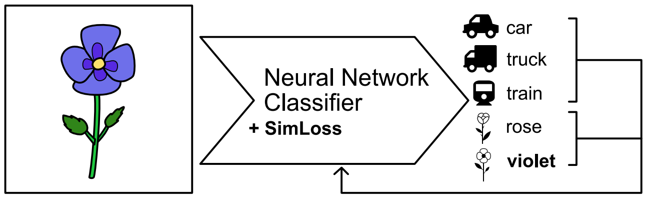


Fig. 1: SimLoss includes knowledge about class relations in the loss function.

class similarities. We show on two tasks, Age Estimation (exploiting class order) and Image Classification (exploiting semantic similarities using word embeddings), that SimLoss can significantly outperform CCE. We also show that tuning the hyper-parameters of both generation techniques influences the model's performance on metrics measuring either more or less specific predictions.

Our contribution is twofold: First, we introduce a drop-in replacement for CCE that incorporates class similarities to support the training of neural network classifiers. Second, we describe two techniques to convert task-specific knowledge into matrices that can be used in the proposed loss function.

2 Related Work

Previous work on including task-specific knowledge in classification is mostly designed for specific use cases, requires modifications to the model architecture or training procedure, or implicitly learns the information while training. Sukhbaatar et al. *implicitly* learn a probability instead of a similarity matrix (we provide an analysis of the relationship in the online material) that indicates the chance of a falsely assigned class label in order to compensate for noise [13]. This, however, requires changes in the network architecture and a special training procedure. An analysis of the relation between probability and similarity based matrices is further analyzed in the online resources. Related to tasks with similar classes are tasks where classes have a taxonomic structure, which is called hierarchical classification. Specifically designed loss functions and/or model architectures use the fact that classes that belong to the same category are more similar than others [1,14]. Izbicki et al. exploit the geospatial relation between areas on earth to automatically geotag input photos [5]. Their model learns to predict a mixture of densities that spread across multiple areas instead of specific classes/areas. A number of task-specific methods try to use the inherent class order of so-called ordinal classification tasks [3,4]. For example, Niu et al. use multiple binary classifications each indicating whether the value is greater than the class value [11]. Model architectures incorporating semantic similarities using word embeddings were shown to usually predict more similar classes if they fail compared to models without similarity information [2,12]. In contrast to the related work, the use of SimLoss does not require special model architectures and works on any common neural network classifier. This makes it easy to explicitly support the training procedure with background knowledge.

3 Similarity Based Loss

Our proposed Similarity Based Loss (SimLoss) is based on the Categorical Cross Entropy (CCE). CCE assumes that only one class is correct and is defined as $L_{\text{CCE}} = -\frac{1}{N}\sum_{i=1}^{N}\log(\boldsymbol{p}_i[y_i])$, where N is the size of the dataset and $\boldsymbol{p}_i[y_i]$ is the probability vector output of the network at the target index y_i for the ith example. To model additional knowledge, SimLoss adds a matrix $\boldsymbol{S}$, which gives

$$L_{\text{SimLoss}} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\sum_{c=1}^{C}\boldsymbol{S}_{y_i,c}\cdot\boldsymbol{p}_i[c]\right), \tag{1}$$

where $\boldsymbol{S}\in[0,1]^{C\times C}$ encodes class relations. $\boldsymbol{S}_{i,j}$ is the similarity between classes i and j. $\boldsymbol{S}_{i,j}=1$ if and only if classes i and j are identical or interchangeable.

SimLoss is equal to CCE if $\boldsymbol{S}=I_c$ (identity matrix). Non zero values lead to smaller losses when the network gives a high score to classes similar to the correct one. For misclassifications, this leads the network to predict similar classes.

Matrix Generation We now propose two techniques to generate the matrix $\boldsymbol{S}$, which explicitly captures background knowledge about class relations. Our techniques allow the modeling of class order and general class similarities.

Class Order: If classes have an inherent order, we can calculate class similarities based on the distance between the class indices. As classes lying next to each other are more similar, we construct the similarity matrix $\boldsymbol{S}$ as follows: Assuming the same distance between neighboring classes, we define the reduction factor $r\in[0,1)$ to be the rate at which the similarity will get smaller given the distance to the correct class. The similarity matrix is then

$$\boldsymbol{S}_{i,j} = r^{|i-j|} \quad \forall i,j\in\{1,\ldots,C\}. \tag{2}$$

The smaller the reduction factor, the faster the entries converge to 0 with increasing distance to the target class. If the reduction factor is set to 0, the matrix becomes the identity, resulting in the CCE loss. The reduction factor is a hyper-parameter of this technique, which can be tuned using a validation dataset to optimize the model for different metrics, as we show in Section 4. As SimLoss is equivalent to CCE when $r=0$ (assuming $0^0=1$), an optimized r will always perform at least as good as CCE unless we overfit.

General Class Similarity: For some classification tasks, a similarity between classes, such as class names, is available or can be defined. Then, we can use an appropriate similarity measure $sim: C\times C\to[0,1]$ that returns the similarity for two classes $i,j\in\{1,\ldots,C\}$ and calculate all entries of the similarity matrix $\boldsymbol{S}$. Such similarity measures can be manual, semi- or fully-automatic. Additionally, we define a lower bound $l\in[0,1)$ as a hyper-parameter that controls the minimal class similarity that should have an impact on the network punishment. We cut all similarities below l and then scale them such that l becomes 0:

$$\boldsymbol{S}_{i,j} = \frac{max(0, sim(i,j)-l)}{1-l}\ \forall i,j\in\{1,...,C\}. \tag{3}$$

Assuming only the diagonal of $\boldsymbol{S}$ are ones, converging $l\to 1$ leads to the CCE loss, as only the ones in the diagonal are preserved by the lower bound cut-off.

4 Experiments

In the following, we compare SimLoss to CCE by applying them to the same neural network model with the same hyper-parameters for Age Estimation and Image Classification. *Age Estimation* is an ordinal classification task with the goal of predicting the age of a person given an image of their face. The classes have an inherent order: two classes are more similar if they represent similar ages. A misclassification is thus less harmful for nearer classes. In *Image Classification*, the goal is to recognize an object shown in an image. Here, we use class name word embeddings to model semantic class similarities. For example, classifying an image of a rose as "violet" is less harmful than classifying it as "truck".

Datasets and Resources For *Age Estimation*, we train neural networks on the UTKFace [15] and AFAD [11] datasets, both containing human face images annotated with their age. For UTKFace, we use all images for ages 1 to 90, while AFAD has 61 age classes. We randomly sample training/validation/test sets using 60/20/20 splits. For *Image Classification*, we use the CIFAR-100 dataset [7]. We also use word embeddings from a word2vec model pretrained on Google News [9] to calculate the semantic similarity between class names. Four class names do not yield a word embedding and are therefore eliminated. Each remaining class has 450 training, 50 validation, and 100 test examples.

Evaluation Metrics To evaluate our method, we employ task-dependent evaluation metrics that focus both on correct predictions and the similarity of predicted and target class. For *Age Estimation:* Accuracy (Acc), Mean Absolute Error (MAE), and Mean Squared Error (MSE). Accuracy captures exact predictions, while MAE and MSE capture the distance to the target class, thus considering class order. *Image Classification:* Accuracy, Superclass Accuracy (SA), and Failed Superclass Accuracy (FSA). Every example in the CIFAR-100 dataset has a main class and a superclass (e.g., classes "rose" and "orchid" have the superclass "flower"). Superclass Accuracy is the fraction of examples that are correctly put into the corresponding superclass. This value is always at least as high as Accuracy, as a correctly assigned class implies the correct superclass. Failed Superclass Accuracy only observes misclassified examples, thus measuring the similarity of misclassifications compared to the target class. A high FSA means that if the model predicts the wrong class, the predicted class is at least similar to the correct class. Accuracy only counts exact predictions, while SA and FSA focus on the semantic similarity of the prediction to the target class.

Generating the Similarity Matrix Since *Age Estimation* has equidistant classes, the similarity matrix can be built using Equation (2) without any modifications. In *Image Classification*, we define the similarity matrix as the cosine similarity $sim_{cos} : w \rightarrow [-1, 1]$ between class name embeddings, where $sim_{cos}(w, w) = 1$. To ensure compatibility with the definition in Section 3, we set $sim(i, j) = \max(0, sim_{cos}(w_i, w_j))$ in Equation (3).

Experimental Setup Since SimLoss is a drop-in replacement for CCE, we investigate the effects of changing the loss function on our example tasks. Recall that we do not focus on task specific models, but rather on the evaluation of SimLoss as a general loss function which can be used on various tasks. Both classification tasks are typical examples for using CCE. For *Age Estimation*, we take the Convolutional Neural Network (CNN) from [11] and change the output size to be the dataset's number of classes. The input images are resized to 60 px by 60 px and the values of all color channels are standardized. We use the softmax function and apply the SimLoss loss function using the similarity matrix introduced above. We study the effect of the reduction factor r by performing grid search for $r \in \{0.0, 0.1, \ldots, 0.9\}$ on the validation set. Optimizing the network using Adam [6] with a learning rate of 0.001 and a batch size of 1024, we employ early stopping [10] with a patience of 10 epochs on the validation MAE. We smooth random differences (e.g., by weight initialization) by averaging over 10 runs. For *Image Classification*, the LeNet CNN [8] is used. Global standardization is applied to the color channels of the input images. We stop early if the Accuracy on the validation set plateaus for 20 epochs of the Adam optimizer with a learning rate of 0.001, and a batch size of 1024. We optimize the matrix generation technique's lower bound $l \in \{0.0, 0.1, \ldots, 0.8, 0.9, 0.99\}$ with grid search and average 10 runs per configuration. $l = 0.99$ makes the loss equivalent to CCE, cutting all similarities except the diagonal.

5 Results

Table 1 shows the resulting mean metrics for the validation and test sets given a reduction factor r for both *Age Estimation* datasets. The best performing reduction factors on the validation and test set are always higher than 0.0, meaning that SimLoss outperforms CCE. Choosing the reduction factor then depends on the metric to optimize for. For UTKFace, a reduction factor of 0.3 leads to the best validation Accuracy, while 0.8 or 0.9 optimize MAE and MSE, respectively. For AFAD, $r = 0.5$ yields the best validation result on Accuracy, while $r = 0.7$ results in the best MAE and MSE. Overall, choosing a smaller reduction factor $r \approx 0.4$ optimizes the Accuracy, while larger $r \approx 0.8$ optimizes MAE and MSE. This is because large r lead to higher matrix values and thus smaller punishments for estimating a class near the correct age. A model optimized for that is favored by metrics that accept approximate matches, such as MAE or MSE.

A Wilcoxon-Signed-Rank-Test with a confidence interval of 5 % shows that optimizing the reduction factor always leads to significant improvements over CCE. Sometimes, however, choosing the reduction factor based on a specific metric also results in a trade-off between the chosen and other metrics.

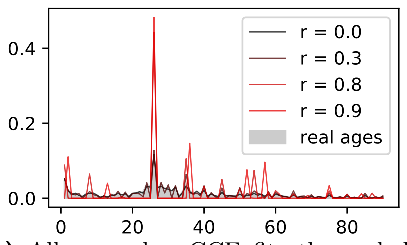
For the *Image Classification* task, Table 2 shows the results for the validation and test set of the CIFAR-100 dataset given a lower bound l. On average, the best performing model always has a lower bound of less than 0.99, again showing that SimLoss outperforms CCE. Also, a statistical test reveals that $l = 0.9$ gives significantly better results on the test set in terms of Accuracy and Superclass Accuracy. Smaller lower bounds tend to reduce the Accuracy as the loss function hardly punishes any misclassification. For $l \approx 1$, the loss is equivalent to CCE, forcing the network to predict the correct class, thus increasing Accuracy. In between, the network is guided to predict the correct class but is also not pun-

Table 1: Validation and test results averaged over 10 runs on UTKFace and AFAD. Accuracy (Acc) is given in percent. Best validation values are written in bold. Statistically significantly different test values are marked by + or −, if they are on average better or worse than CCE (i.e. $r = 0.0$).

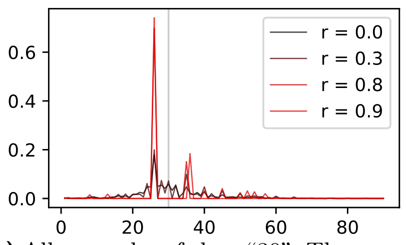
	UTKFace						AFAD					
	Validation			Test			Validation			Test		
r	Acc	MAE	MSE	Acc	MAE	MSE	Acc	MAE	MSE	Acc	MAE	MSE
0.0	15.23	7.09	122.12	14.47	7.39	131.65	11.17	4.05	32.61	11.22	4.10	33.64
0.1	15.43	7.06	119.87	14.48	7.29	127.18	11.21	4.06	32.75	11.30	4.10	33.73
0.2	15.94	7.06	121.28	14.57	7.27	127.13	11.40	4.09	33.52	11.37	4.15^-	34.60^-
0.3	**16.25**	6.95	117.67	15.17^+	7.19^+	125.70	11.34	4.10	33.53	11.38^+	4.16^-	34.53^-
0.4	16.13	6.95	117.52	15.46^+	7.18^+	125.74	11.33	4.10	33.44	11.45^+	4.16^-	34.56^-
0.5	16.10	6.89	115.59	15.09	7.18^+	123.94	**11.44**	4.06	33.02	11.49^+	4.13	34.21
0.6	15.62	6.83	112.85	14.34	7.09^+	120.34^+	11.26	4.01	31.99	11.31	4.05^+	32.84^+
0.7	14.39	6.79	110.12	13.07	7.08^+	121.19^+	11.22	**3.95**	**31.17**	11.11	4.02^+	32.36^+
0.8	13.50	**6.74**	108.80	12.57^-	7.01^+	117.99^+	8.58	4.58	38.69	8.55^-	4.64	39.78
0.9	9.69	6.90	**106.23**	9.16^-	7.18^+	117.62^+	6.55	5.09	44.87	6.47^-	5.15^-	45.82^-

ished severely for misclassifications of similar classes. This improves Superclass Accuracy, which pays attention to more similar classes.

Analysis To understand the effect of SimLoss, we focus on Age Estimation whose one dimensional classes are easy to visualize. We compare the best models for UTKFace trained using SimLoss and CCE, i.e. $r \in \{0.0, 0.3, 0.8, 0.9\}$. For each r, we plot the mean output distribution for all examples in the dataset as well as the real age distribution, which is shown in Figure 2a. CCE ($r = 0.0$) resembles the real age distribution the best, while higher reduction factors tend to aggregate groups of multiple age classes. With a higher reduction factor,



a) All examples. CCE fits the real data distribution the best.



b) All examples of class "30". The grey line indicates the target age.

Fig. 2: Mean probability distribution output for different r. High reduction factors lead the network to choose only few representative classes.

Table 2: Validation and test results over 10 runs with early stopping on the modified CIFAR-100 dataset. Best validation values are written in bold. Statistically significantly different test values are marked by + or −, if they are on average better or worse than CCE (i.e. $l = 0.99$).

l	Validation			Test		
	Accuracy	SA	FSA	Accuracy	SA	FSA
0.99	46.89 %	55.78 %	16.73 %	39.51 %	49.22 %	16.05 %
0.90	**47.42 %**	56.32 %	16.95 %	40.15 %$^+$	49.93 %$^+$	16.36 %
0.80	46.37 %	55.38 %	16.80 %	39.49 %	49.32 %	16.22 %
0.70	46.95 %	55.92 %	16.90 %	39.86 %	49.63 %	16.25 %
0.60	47.28 %	**56.44 %**	17.39 %	40.00 %	50.00 %	16.67 %$^+$
0.50	46.36 %	56.18 %	18.28 %	39.26 %	49.40 %	16.70 %$^+$
0.40	38.03 %	50.58 %	20.28 %	32.18 %$^-$	44.58 %$^-$	18.30 %$^+$
0.30	28.65 %	43.76 %	**21.18 %**	24.43 %$^-$	38.90 %$^-$	19.13 %$^+$
0.20	21.66 %	37.97 %	20.80 %	18.54 %$^-$	33.68 %$^-$	18.58 %$^+$
0.10	16.40 %	31.68 %	18.31 %	14.25 %$^-$	28.70 %$^-$	16.85 %
0.00	2.80 %	8.37 %	5.77 %	2.53 %$^-$	8.06 %$^-$	5.71 %$^-$

the number of spikes decreases and the distances between them increase: The model chooses representative classes to which it mainly distributes the output probability mass. This becomes apparent in Figure 2b, where we plot the mean output distribution for all examples of age 30. The network with $r = 0.9$ focuses its probability output to the two nearest representative classes, in this case "26" and "35". The Accuracy of the network decreases, as the output probability mass is not on the correct class, but the distance of the prediction to the correct class is smaller than for CCE. Representative classes are apparently chosen such that frequent items receive more probability mass from the model. A higher reduction factor therefore leads to a coarser class selection. This can be explained by the optimization objective of the loss function. The loss should be smaller for misclassifications of similar classes than for dissimilar classes. Representing multiple similar classes as one class and predicting it more often for similar classes does not lead to the smallest possible loss value. However, the loss gets smaller compared to predicting dissimilar classes, as the punishment should be smaller for classifying a similar class. In the case of Age Estimation, predicting an age that lies close to the correct age will decrease the Accuracy, but perform better than CCE on MAE and MSE. In Image Classification, selecting one or multiple representative classes leads to smaller Accuracy but to higher Superclass Accuracy and Failed Superclass Accuracy than CCE. Higher similarities in the matrix thus guide the network to make coarser predictions, improving metrics that accept predictions of similar classes. The results from Section 4 also show that keeping the loss near CCE by choosing the similarity matrix conservatively can improve on specific prediction metrics such as Accuracy as well.

6 Conclusion

In this work, we have presented SimLoss, a modified Categorical Cross Entropy loss function that incorporates background knowledge about class relations in form of class similarities. We have introduced two techniques to prepare similarity matrices to exploit class order and general class similarity that can be used to significantly improve the performance of neural network classifiers on different metrics. Also, SimLoss helped with predicting more similar classes if the model misclassified an example. In our analysis, we found that SimLoss forced the model to focus on choosing representative classes. The number of representative classes can be implicitly tuned by a hyper-parameter. While finding the best hyper-parameter and similarity metric can be computationally expensive and non-trivial, SimLoss can incorporate arbitrary similarity metrics into a classifier.

References

1. Cesa-Bianchi, N., Gentile, C., Zaniboni, L.: Incremental algorithms for hierarchical classification. Journal of Machine Learning Research **7**(Jan), 31–54 (2006)
2. Frome, A., Corrado, G.S., Shlens, J., Bengio, S., Dean, J., Mikolov, T., et al.: Devise: A deep visual-semantic embedding model. In: NIPS (2013)
3. Fu, Y., Huang, T.S.: Human age estimation with regression on discriminative aging manifold. IEEE Transactions on Multimedia **10**(4), 578–584 (2008)
4. Guo, G., Mu, G., Fu, Y., Huang, T.S.: Human age estimation using bio-inspired features. In: CVPR. IEEE (2009)
5. Izbicki, M., Papalexakis, E.E., Tsotras, V.J.: Exploiting the earth's spherical geometry to geolocate images. In: ECML-PKDD (2019)
6. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
7. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Tech. rep., Citeseer (2009)
8. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
9. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
10. Morgan, N., Bourlard, H.: Generalization and parameter estimation in feedforward nets: Some experiments. In: NIPS (1990)
11. Niu, Z., Zhou, M., Wang, L., Gao, X., Hua, G.: Ordinal regression with multiple output cnn for age estimation. In: CVPR (2016)
12. Norouzi, M., Mikolov, T., Bengio, S., Singer, Y., Shlens, J., Frome, A., Corrado, G.S., Dean, J.: Zero-shot learning by convex combination of semantic embeddings. arXiv preprint arXiv:1312.5650 (2013)
13. Sukhbaatar, S., Bruna, J., Paluri, M., Bourdev, L., Fergus, R.: Training convolutional networks with noisy labels. arXiv preprint arXiv:1406.2080 (2014)
14. Wu, C., Tygert, M., LeCun, Y.: Hierarchical loss for classification. arXiv preprint arXiv:1709.01062 (2017)
15. Zhang, Z., Song, Y., Qi, H.: Age progression/regression by conditional adversarial autoencoder. In: CVPR (2017)
16. Zhang, Z., Sabuncu, M.: Generalized cross entropy loss for training deep neural networks with noisy labels. In: NIPS (2018)

Appendix

Relation between Similarity and Probability-based Matrices in SimLoss

Some works use a loss function similar to our proposed SimLoss. Instead of similarities, the matrix $\boldsymbol{S}$ consists of probabilities, such that each row sums to one [5,13]. We will call this loss L_{prob}. We discuss the relation between both loss versions — similarity versus probability matrix — in this appendix.

We can show that both similarities and probabilities in the matrix lead to the same gradients: We can transform our loss L_{SimLoss} into L_{prob} by dividing each similarity matrix entry by the row's sum. This loss depends on the network's output — the probability distribution p — as well as the corresponding target class indices y. It can be written as:

$$
\begin{aligned}
L_{\text{prob}} &= -\frac{1}{N}\sum_{i=1}^{N}\log\left(\sum_{c=1}^{C}\frac{1}{\sum_{c'=1}^{C}\boldsymbol{S}_{y_i,c'}}\boldsymbol{S}_{y_i,c}\cdot\boldsymbol{p}_i[c]\right)\\
&= -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{\sum_{c'=1}^{C}\boldsymbol{S}_{y_i,c'}}\sum_{c=1}^{C}\boldsymbol{S}_{y_i,c}\cdot\boldsymbol{p}_i[c]\right)\\
&= -\frac{1}{N}\sum_{i=1}^{N}\left[\log\left(\sum_{c=1}^{C}\boldsymbol{S}_{y_i,c}\cdot\boldsymbol{p}_i[c]\right)-\log\left(\sum_{c'=1}^{C}\boldsymbol{S}_{y_i,c'}\right)\right]\\
&= -\frac{1}{N}\sum_{i=1}^{N}\left[\log\left(\sum_{c=1}^{C}\boldsymbol{S}_{y_i,c}\cdot\boldsymbol{p}_i[c]\right)\right]+\frac{1}{N}\sum_{i=1}^{N}\left[\log\left(\sum_{c'=1}^{C}\boldsymbol{S}_{y_i,c'}\right)\right]\\
&= L_{\text{SimLoss}}+\frac{1}{N}\sum_{i=1}^{N}\left[\log\left(\sum_{c'=1}^{C}\boldsymbol{S}_{y_i,c'}\right)\right] \quad .
\end{aligned}
$$

The second summand leads to different loss function values for both matrices. It does not depend on the probability output p. Therefore, when calculating the gradients with respect to p, this term becomes zero:

$$\frac{\partial}{\partial p}L'_{\text{SimLoss}} = \frac{\partial}{\partial p}L_{\text{SimLoss}} \quad .$$

Both L_{SimLoss} and L_{prob} yield the same gradients when optimizing the model. If the largest values in the matrices are on the diagonal, both matrix variants will have the same parameters when reaching the global optimum [16]. Even though both methods theoretically lead to the same results, our method is less restrictive since it does not require a probability distribution per row. For example, while similarities can be calculated for each class pair independently, a probability distribution needs to be normalized over all values in the row prior to use. For tasks with a large number of classes, the similarity matrix might not need to be stored but could be calculated on the fly, while probabilities would cause larger computational costs. Especially on edge devices with very limited memory, this is an advantage of our method.

Another advantage of similarities compared to probabilities is that, because the diagonal of the similarity matrix consists of ones, a value of zero can be reached by the loss function, making the loss value more interpretable. A loss value of zero always means that the probability mass of the neural network output vector is put into the correct class, even if there are similar classes. Normalizing such a matrix to ensure probability distributions per row would always yield larger loss values, even if the correct class is predicted with 100 % probability. Therefore, L_{SimLoss} always has a lower bound of zero, which gives an interpretable impression of the training status. In a probability matrix based loss, such an interpretation is not given as the lower bound of the loss depends on the probability distributions in the matrix rows. A loss value of zero can only be achieved when using the identity matrix. This, however, would be equivalent to Categorical Cross Entropy and would not allow for including background knowledge into the model.