A. Model Architectures and Experimental Details

Tabs. 4, 6 and 9 show the architectures of the auto-encoders used for the three different data sets. Tabs. 5, 7, 10 show the architectures of the classifiers, generator and discriminators. Tab. 8 shows the structure of the residual generators and discriminators. Tab. 3 shows the number of training iterations used for each GAN and data set. Wherever possible we use the same training parameters as the author of the accompanying paper. Therefore we follow [6] and [25] to train all the GANs. We only use a different amount of training iterations depending on the data set. Futhermore, we follow [22] to train the classifier used for *CIFAR-10 - ten classes*.

Table 3: Number of training iterations for the different GANs and data sets.

GAN	MNIST		CIFAR-10		CelebA	SVHN	LSUN
	two	ten	two	ten			
DCGAN	40k	100k	100k[1]	200k	—	—	—
WGAN-GP	40k	100k	200k	200k	100k	100k	100k
Res-WGAN-GP	—	—	100k	100k	—	—	—
Res-WGAN-CT	—	—	100k	100k	—	—	—

A.1. MNIST - two classes

For binary digit classification we train a linear model with cross entropy loss. We train the model for 10 epochs using the Adam optimizer [11] with a batch size of 10 and learning rate of 0.001. We train the Wasserstein GAN [6] with gradient penalty to synthesize only the digits 5 & 7. Similarly, we train the auto-encoder for 40 epochs with a batch size of 100 using the Adam optimizer with a learning rate of 0.001 using only the digits 5 & 7. For the binary experiments we decrease the number of channels in the auto encoder (see Tab. 4) by a factor of three.

A.2. CIFAR-10 - two classes

For *CIFAR-10 - two classes* we train a linear model with cross entropy loss. We train the model for 10 epochs using the Adam optimizer [11] with a batch size of 50 and learning rate of 0.001. We train the Wasserstein GAN [6] with gradient penalty to synthesize only the two target classes. Similarly, we train the auto-encoder for 100 epochs with a batch size of 128 using the Adam optimizer with a learning rate of 0.001 using only the images with the two target classes.

[1]We observe mode collapse for *CIFAR-10 - two classes* using DCGAN when training the GAN for more than 100k iterations.

A.3. MNIST - ten classes

For ten digit classification we use LeNet ([14]) with cross entropy. We train the model for 10 epochs using the Adam optimizer with a batch size of 50 and learning rate of 0.001. We train the Wasserstein GAN [6] with gradient penalty and the auto-encoder for 40 epochs with a batch size of 100 using the Adam optimizer with a learning rate of 0.001.

A.4. CIFAR-10 - ten classes

Using all classes of CIFAR-10 complicates the classification task and we require a deep model to achieve close to state-of-the-art results. Therefore, we use the All-CNN model proposed by [22] with a reported classification error of 9.08%. We use the proposed architectures and training strategies and use stochastic gradient descent with constant momentum of 0.9 and a learning rate of 0.01 that we decay by a factor of 10 at the 130th and the 140th epoch. We train the model for 150 epochs with a batch size of 128 without data augmentation and report a classification error of 11.8%. The All-CNN contains $\sim$1.4 million different parameters. Hence, we require larger initial training sets than for the previous models. Thus we include 100 randomly selected images per class. We add 1000 samples to the data set every AL cycle until the budget of 30k samples is reached. We generate ten times a batch containing 100 samples because optimizing for all samples at the same time is unfeasible. In addition to the previous experiments we test a traditional layered and a residual structure for the GAN. We train both with gradient penalty with or without a soft consistency term.

A.5. CelebA

For classification we use the CNN presented in Tab. 10. We use the Adam optimizer with a learning rate of 0.001 and a batch size of 50 and train for 30 epochs. We start active learning with 100 labelled samples, where the number of samples per class corresponds to the data distribution. We train the auto-encoder for 100 epochs with a batch size of 64 using the Adam optimizer with a learning rate of 0.0001.

A.6. SVHN

For classification we use the Conv-Small CNN proposed by Miyato*et al.* [16] and use the auto encoder and GAN architectures presented in Tabs. 6 and 7. We use the Adam optimizer with a learning rate of 0.001 and a batch size of 128 and train for 120 epochs. We start active learning with 1000 labelled samples, where the number of samples per class corresponds to the data distribution. We train the auto-encoder for 100k iterations with a batch size of 64 using the Adam optimizer with a learning rate of 0.0001.

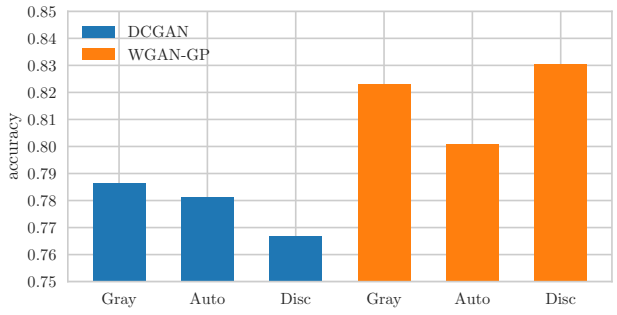


Figure 7: Comparison of the agreement accuracy between manual annotations and matched labels. The matching strategies employed in ASAL allow to select similar images from the pool and compare these labels to the manual annotations. For *MNIST - two classes* the agreement for WGAN-GP is higher than for DCGAN.

A.7. LSUN

For classification we use the Conv-Small CNN proposed by Miyato*et al*. [16] because the architecture is designed for 32×32 color images we change the first layer to use 5×5 convolution kernels with stride two. The auto encoder and GAN architectures are presented in Tabs. 9 and 10. We use the Adam optimizer with a learning rate of 0.001 and a batch size of 128 and train for 120 epochs. We start active learning with 1000 labelled samples, where the number of samples per class corresponds to the data distribution. We train the auto-encoder for 100k iterations with a batch size of 64 using the Adam optimizer with a learning rate of 0.0001.

B. Additional Results: MNIST - two classes

Fig. 8 shows the test accuracy of ASAL for different uncertainty scores and GANs. Fig. 9 and 10 shows the label distribution during active learning for ASAL an classical methods. Fig. 11 shows the entropy of newly added samples.

B.1. Agreement of Manual Annotations and Matched Labels

Instead of manually annotating images we propose to select similar images from the pool and ask for labels of these images. Similar images might show an object of the same class, have similar surrounding, colors, size or share other features. Thus, we compare the agreement of the manual class annotations of the generated images with the matched images, using the three different strategies. We use 1300 generated samples for each GAN, annotate the images manually and retrieve the closest match with the corresponding label from the pool. We assume that the final model will be measured on an almost evenly distributed test set similar to MNIST and USPS. However, the test set for this experiment contains the generated samples with manual annotations and the GAN may generate samples with unevenly distributed label frequency. Thus, we compute the accuracy for each class independently and average these values subsequently to obtain the final score.

Fig. 7 shows that the agreement is higher for ASAL strategies using WGAN-GP than DCGAN. Furthermore, we observe that the matching based on gray values achieves the highest agreement. Similarly, Figs. 8a and 8b show best performance for ASAL-Gray.

C. Additional Results: MNIST - ten classes

Fig. 12 shows the test accuracy of ASAL for different GANs. Fig. 15 shows the label distribution during active learning for ASAL an classical methods. Fig. 13 shows the entropy of newly added samples. Fig. 16 shows a comparison of two different methods: (i) randomly sub sampling the pool and scanning this subset for the sample with maximum entropy and (ii) using ASAL to retrieve more samples than required and scan this subset for the samples with maximum entropy. Fig. 16 shows, that for a fixed subset size, ASAL combined with uncertainty sampling retrieves always higher entropy samples than random sampling combined with uncertainty sampling. In addition, ASAL achieves the best classification accuracy and approaches uncertainty sampling faster.

D. Additional Results: CIFAR - two classes

Fig. 19 shows the test accuracy of ASAL for different GANs. Fig. 14, 17 and 18 shows the label distribution during active learning for ASAL an classical methods. For CIFAR-10, we do not indicate the true label distribution by a tick because the validation set contains the same number of samples for each class. Fig. 20 shows the entropy of newly added samples.

E. Additional Results: CIFAR - ten classes

Fig. 21 shows the test accuracy of ASAL for different GANs. Fig. 23 and 24 shows the label distribution during active learning for ASAL and classical methods. For CIFAR-10, we do not indicate the true label distribution by a tick because the validation set contains the same number of samples for each class. Fig. 22 shows the entropy of newly added samples.

E.1. Discussion of results on CIFAR-10 - ten classes

Fig. 24 shows that maximum entropy sampling includes most frequently images showing cats, exactly one of the labels least frequent when using ASAL with auto-encoder

features. We observe that using F_{CLS} leads to a similar distribution although less distinctive. Therefore, we conclude that instead of generating images of cat, the generator produces images of various classes and moves them close to the decision boundary to increase the entropy. However, note that truck and automobile are among the least frequent classes in any experiment and we conclude that the sample generating process is aware that these classes lead to small entropy and produces samples showing other classes.

F. Matching Strategy Visualization

Figs. 25, 26, 27, 28 show examples of generated images of the same active learning cycle and the corresponding matches. All images are generated using WGAN-GP and the maximum entropy score. The generated images are not manually annotated. The moderate quality of the generated CIFAR-10 images prevents confidently annotating the images. Instead, *n.a.* indicates that the manual annotation is missing.

Table 4: Auto-encoder architecture for ASAL on MNIST.

Encoder	**Decoder**
Input: $28 \times 28 \times 1$	Input: $4 \times 4 \times 12$
5×5 conv: 3, stride=2	5×5 deconv: 6
leakyReLU	ReLU
5×5 conv: 6, stride=2	5×5 deconv: 3
leakyReLU	ReLU
5×5 conv: 12, stride=2	5×5 deconv: 1
leakyReLU	sigmoid

Table 5: Model architectures for ASAL on MNIST.

Classifier	**Generator**	**Discriminator**
Input: $28 \times 28 \times 1$	Input: $z \sim \mathcal{N}(0, 1)$: 128	Input: $28 \times 28 \times 1$
5×5 conv: 32	linear: 128×4096	5×5 conv: 64, stride=2
ReLU, Maxpool 2×2	ReLU	leakyReLU
5×5 conv: 64	5×5 deconv: 128	5×5 conv: 128, stride=2
ReLU, Maxpool 2×2	ReLU	leakyReLU
linear: 3136×1024	5×5 deconv: 64	5×5 conv: 256, stride=2
ReLU, Dropout: 0.5	ReLU	leakyReLU
linear: 1024×10	5×5 deconv: 64	linear: 3136×1
	sigmoid	

Table 6: Auto-encoder architecture for ASAL on CIFAR-10 and SVHN.

Encoder	**Decoder**
Input: $32 \times 32 \times 3$	Input: $4 \times 4 \times 16$
3×3 conv: 64, Batch norm	3×3 deconv: 32
ReLU, Maxpool 2×2	Batch norm, ReLU
3×3 conv: 32, Batch norm	3×3 deconv: 64
ReLU, Maxpool 2×2	Batch norm, ReLU
3×3 conv: 16, Batch norm	3×3 deconv: 3
ReLU, Maxpool 2×2	Batch norm, sigmoid

Table 7: Model architectures for ASAL on CIFAR-10 and SVHN (Generator and Discriminator).

Classifier	**Generator**	**Discriminator**
Input: $32 \times 32 \times 3$	Input: $z \sim \mathcal{N}(0, 1)$: 128	Input: $32 \times 32 \times 3$
3×3 conv: 96	linear: 128×8192	5×5 conv: 128, stride=2
ReLU	Batch norm, ReLU	leakyReLU
3×3 conv: 96	5×5 deconv: 256	5×5 conv: 256, stride=2
ReLU	Batch norm, ReLU	leakyReLU
3×3 conv: 96, stride=2	5×5 deconv: 128	5×5 conv: 512, stride=2
ReLU, dropout=0.5	Batch norm, ReLU	leakyReLU
3×3 conv: 192	5×5 deconv: 3	linear: 8192×1
ReLU	Tanh	
3×3 conv: 192		
ReLU		
3×3 conv: 192, stride=2		
ReLU, dropout=0.5		
3×3 conv: 192		
ReLU		
1×1 conv: 192		
ReLU		
1×1 conv: 10		
global average pool		

Table 8: Residual GAN architectures for ASAL on CIFAR-10.

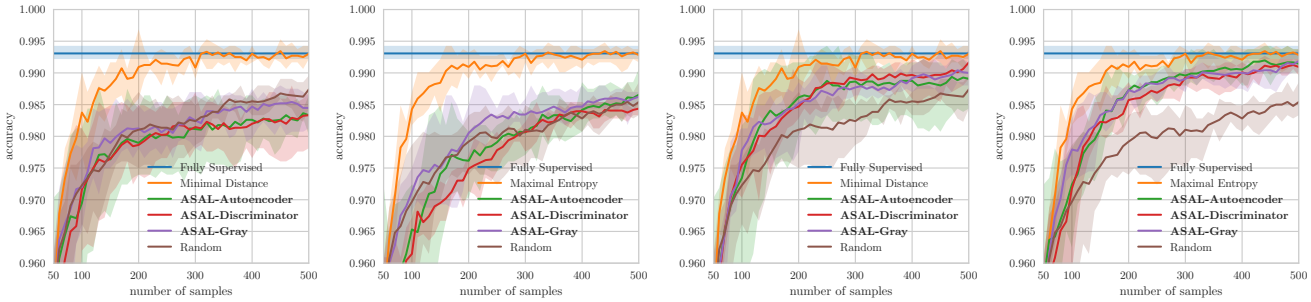
Generator	**Discriminator**
Input: $z \sim \mathcal{N}(0, 1)$: 128	Input: $32 \times 32 \times 3$
linear: 128×2048	$[3 \times 3] \times 2$ ResidualBlock: 128
	Down=2
$[3 \times 3] \times 2$ ResidualBlock: 128	$[3 \times 3] \times 2$ ResidualBlock: 128
Up=2	Down=2
$[3 \times 3] \times 2$ ResidualBlock: 128	$[3 \times 3] \times 2$ ResidualBlock: 128
Up=2	
$[3 \times 3] \times 2$ ResidualBlock: 128	$[3 \times 3] \times 2$ ResidualBlock: 128
Up=2	ReLU, MeanPool
3×3 conv: 3	linear: 128×1
Tanh	

Table 9: Auto-encoder architecture for ASAL on CelebA and LSUN.

Encoder	Decoder
Input: $64 \times 64 \times 3$	Input: $4 \times 4 \times 16$
5×5 conv: 128, stride=2	5×5 deconv: 32
Batch norm, ReLU	Batch norm, ReLU
5×5 conv: 64, stride=2	5×5 deconv: 64
Batch norm, ReLU	Batch norm, ReLU
5×5 conv: 32, stride=2	5×5 deconv: 128
Batch Norm, ReLU	Batch norm, ReLU
5×5 conv: 16, stride=2	5×5 deconv: 3
	Tanh

Table 10: Model architectures for ASAL on CelebA and LSUN (Generator and Discriminator).

Classifier	Generator	Discriminator
Input: $64 \times 64 \times 3$	Input: $z \sim \mathcal{N}(0, 1)$: 128	Input: $64 \times 64 \times 3$
3×3 conv: 16	linear: 128×4096	5×5 conv: 128, stride=2
ReLU, Maxpool 2×2	Batch norm, ReLU	leakyReLU
3×3 conv: 32	5×5 deconv: 256	5×5 conv: 256, stride=2
ReLU, Maxpool 2×2	Batch norm, ReLU	leakyReLU
3×3 conv: 64	5×5 deconv: 128	5×5 conv: 512, stride=2
ReLU, Maxpool 2×2	Batch norm, ReLU	leakyReLU
linear: 4096×1024	5×5 deconv: 64	5×5 conv: 512, stride=2
ReLU, Dropout 0.5	Batch norm, ReLU	leakyReLU
linear: 1024×1	5×5 deconv: 3	linear: 8192×1
	Tanh	



(a) *Minimum distance* with Hinge loss and DCGAN. (b) *Maximum entropy* with cross-entropy loss and DCGAN. (c) *Minimum distance* with Hinge loss and WGAN-GP. (d) *Maximum entropy* with cross-entropy loss and WGAN-GP.

Figure 8: Test accuracy on *MNIST - two classes* of a fully supervised model, for random sampling, uncertainty sampling and different ASAL using different GANs, uncertainty measures and loss functions. ASAL with WGAN-GP (bottom) clearly exceed the performance of ASAL using DCGAN (top). Maximum entropy sampling and using the cross entropy loss lead to the setup (8d) that approaches the fully-supervised model with the fewest samples and reaches the smallest gap for all ASAL using 500 labelled samples.

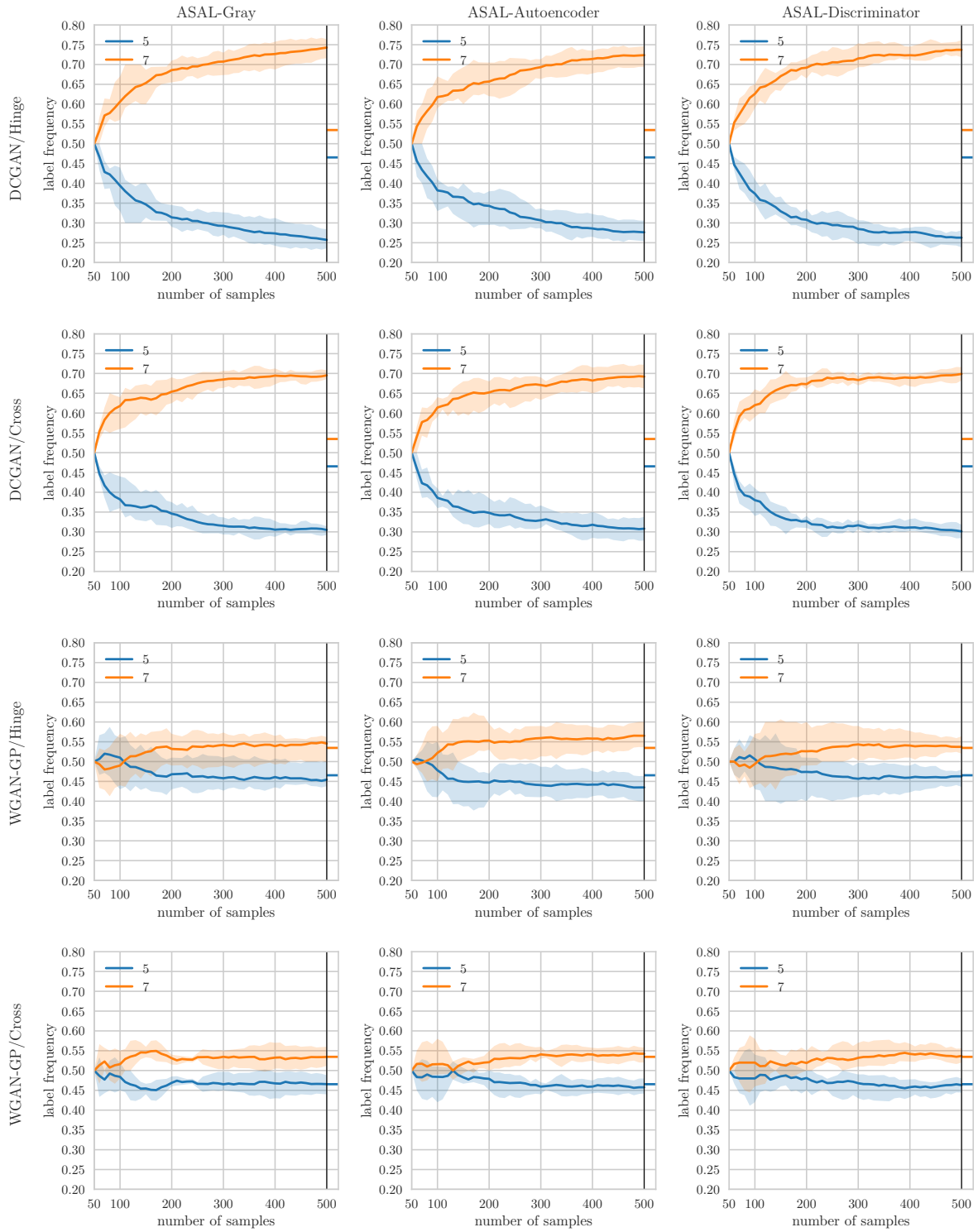
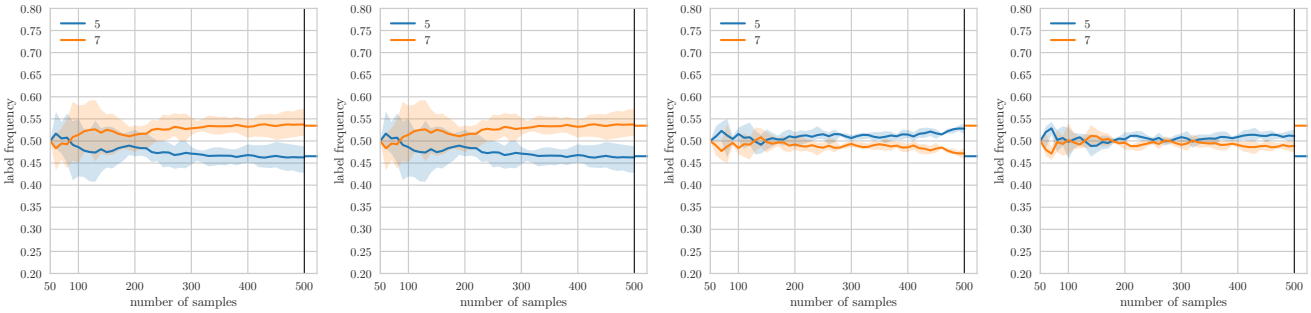
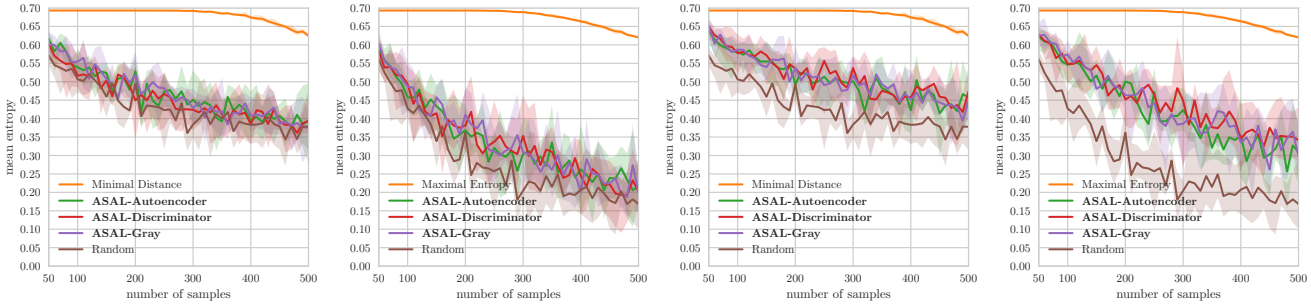


Figure 9: Label distribution for active learning using different matching strategies, uncertainty measures and GANs for *MNIST - two classes*. The ticks on the right show the true label distribution in the pool. ASAL using WGAN-GP (third and fourth row) reaches a label distribution of the training data that is similar to the true label distribution in the pool. Conversely, ASAL using DCGAN (first and second row) leads to a training set that contains almost three times as many images with the digit 7 than digit 5. Most likely, the DCGAN is responsible for this behaviour because we already observed that it produces the digit 7 more frequently than the digit 5.



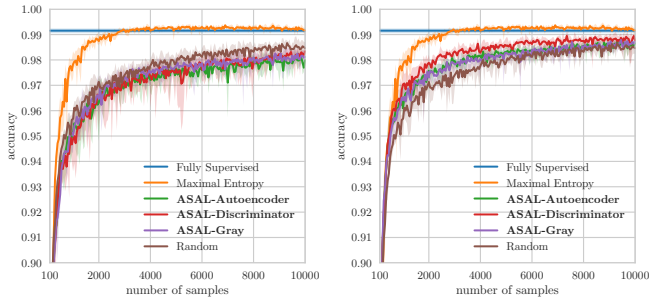
(a) Random sampling with Hinge loss.(b) Random sampling with cross-entropy loss.(c) *Minimum distance* sampling with Hinge loss.(d) *Maximum entropy* sampling with cross-entropy loss.

Figure 10: Label distribution for uncertainty sampling using maximum entropy and random sampling for *MNIST - two classes* using different uncertainty measures and loss functions. The tick on the right show the true label distribution in the pool. The label distribution of the training set, assembled with random sampling (top), converges to the true label distribution of the pool. Conversely, uncertainty sampling leads to a training set that contains more frequently the label 5 than 7 compared to the pool that contains 7 more frequently. Apparently, images with the digit 5 lead to higher uncertainty of the used classifier.



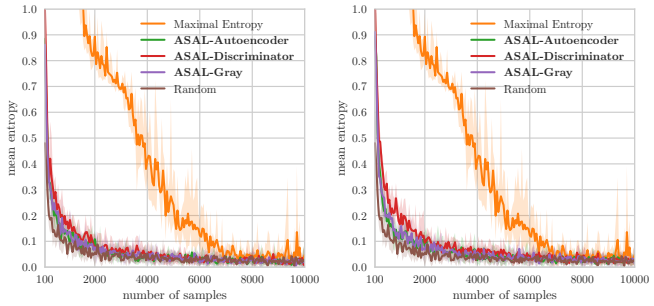
(a) *Minimum distance* with Hinge loss and DCGAN.(b) *Maximum entropy* with cross-entropy loss and DCGAN.(c) *Minimum distance* with Hinge loss and WGAN-GP.(d) *Maximum entropy* with cross-entropy loss and WGAN-GP.

Figure 11: Average entropy of images that are selected and added to the training set for *MNIST - two classes* using different GANs, uncertainty measures and loss functions. All figures show that ASAL selects samples from the pool that have a higher entropy than randomly sampled images. However, maximum entropy sampling and WGAN-GP (11d) lead to the largest entropy gap between selected and randomly sampled images. Maximum entropy sampling (right column) results in smaller average entropy of the classifier than minimum distance sampling (left column) because we use the cross-entropy loss that directly optimizes for small entropy, opposed to the hinge loss that minimizes the distance to the separating hyper-plane.



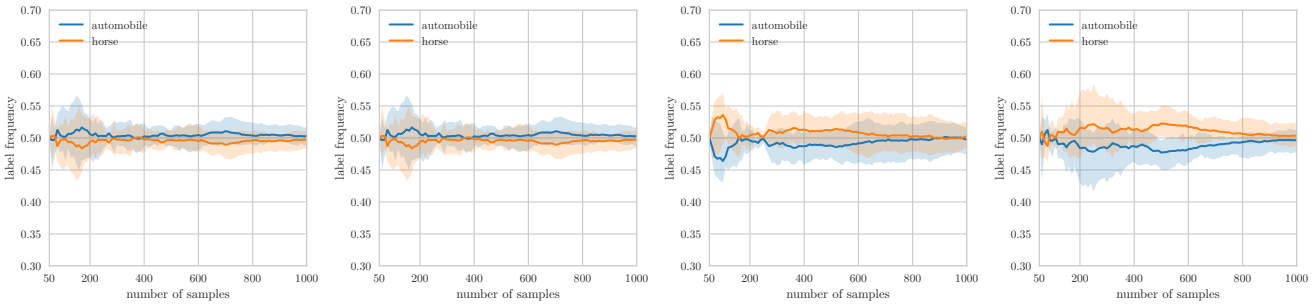
(a) DCGAN. (b) WGAN-GP.

Figure 12: Test accuracy on *MNIST - ten classes* of a fully supervised model, for random sampling, uncertainty sampling and different ASALs using two different GANs. Selecting new images using random samples exceeds the performance of the proposed strategy when using the DCGAN. However, replacing the DCGAN with the WGAN-GP enables outperforming random sampling. ASAL-Discriminator achieves the best quality.



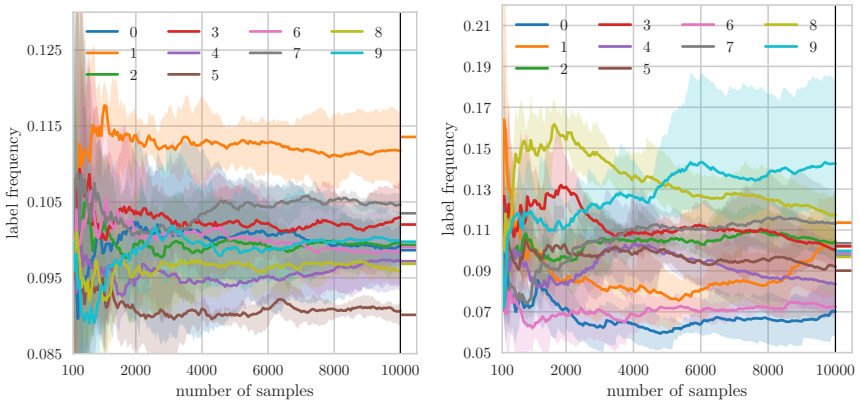
(a) DCGAN. (b) WGAN-GP.

Figure 13: Average entropy of images that are selected and added to the training set for *MNIST - ten classes* using different GANs. Both figures show that at the beginning ASAL selects images with higher entropy than random sampling. In average WGAN-GP leads to a larger gap than DCGAN. However, this gap rapidly shrinks when increasing the training set.



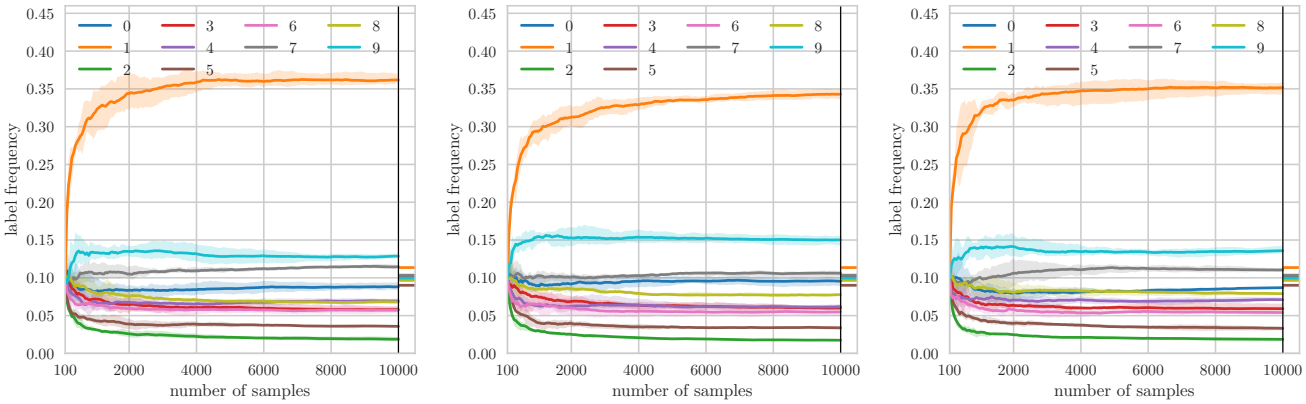
(a) Random sampling with Hinge loss. (b) Random sampling with cross-entropy loss. (c) *Minimum distance* sampling with Hinge loss. (d) *Maximum entropy* sampling with cross-entropy loss.

Figure 14: Label distribution for uncertainty sampling using maximum entropy and random sampling for *CIFAR-10 - two classes* using different uncertainty measures and loss functions.The label distribution of the training set of all strategies converges to the true label distribution of the pool. However, in average over all active learning iterations the training set of the uncertainty sampling strategies most frequently contained the images with the label horse.



(a) Random sampling.

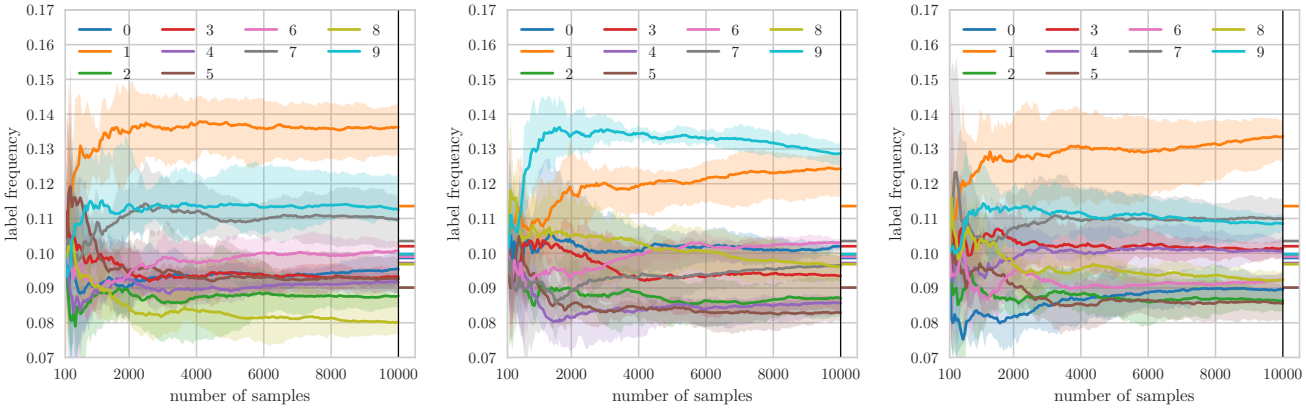
(b) Maximum entropy sampling.



(c) ASAL-Gray with DCGAN.

(d) ASAL-Autoencoder with DCGAN.

(e) ASAL-Discriminator with DCGAN.



(f) ASAL-Gray with WGAN-GP.

(g) ASAL-Autoencoder with WGAN-GP.

(h) ASAL-Discriminator with WGAN-GP.

Figure 15: Label distribution for uncertainty sampling using maximum entropy, random sampling and active learning using different matching strategies and GANs for *MNIST - ten classes*. The tick on the right show the true label distribution in the pool. Note the different scaling of the y-axis. Random sampling converges to the true label distribution in the pool and maximum entropy sampling leads to a training set with a higher ration of certain digits (7,8,9) or lower (0,1,4,6) than the pool. Similarly, ASAL using WGAN-GP (bottom row) selects certain digits more frequently than others. Conversely, ASAL using DCGAN (top row) leads to a training set that contains 30% images with the digit 1. Most likely, the DCGAN is responsible for this behaviour because we already observed that it produces the digit 1 more frequently than any other digit.

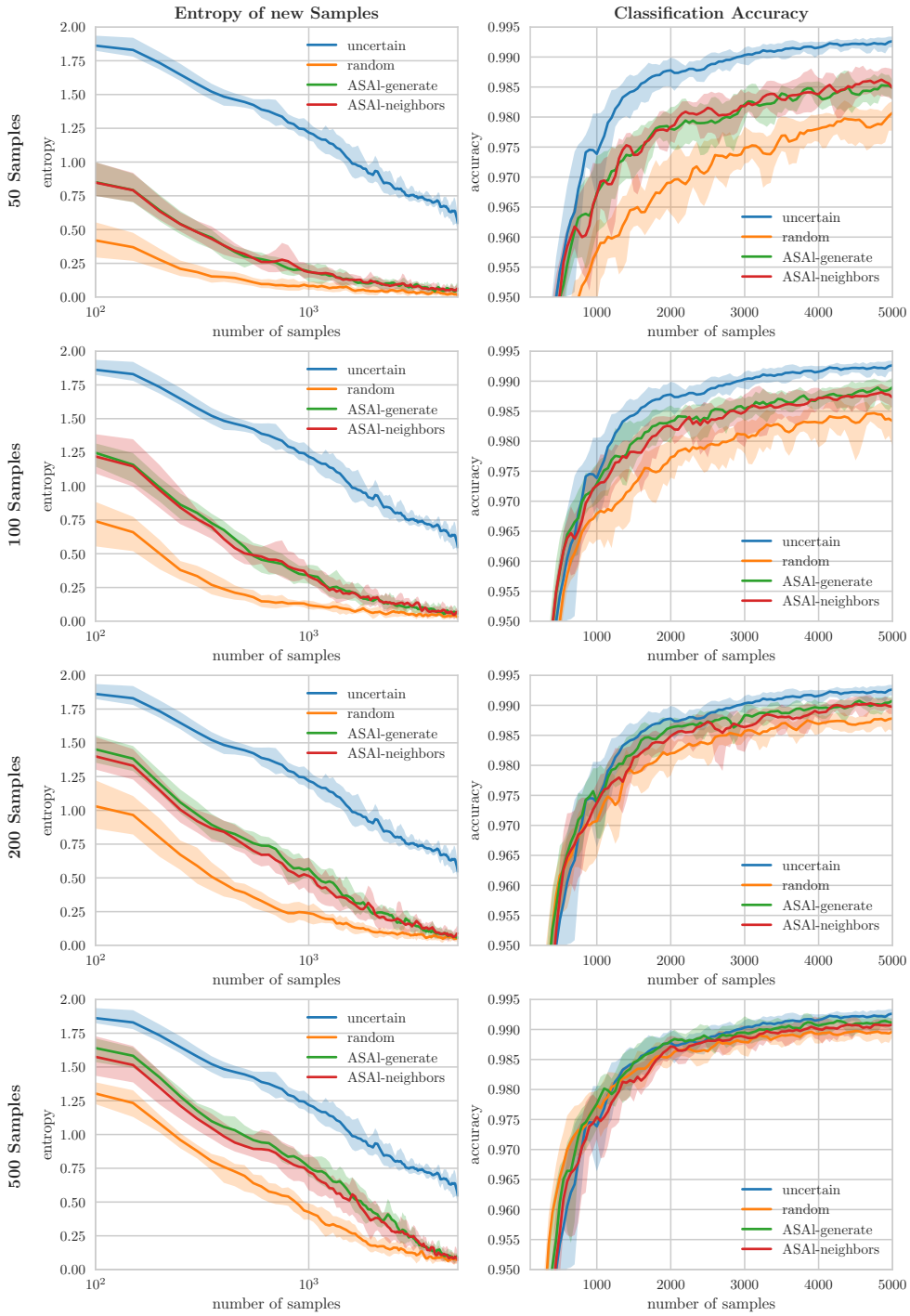


Figure 16: Comparison of random sampling combined with uncertainty sampling and ASAL combined with uncertainty sampling. Instead of selecting only 50 random samples and label all of them, we randomly build a small subset containing a few hundred samples from the pool and retrieve the 50 most uncertain samples, we denote this setting as *random*. For *ASAL-generate*, we generate more than the required 50 samples, match all of these samples and select the 50 most uncertain among all matched real samples. For *ASAL-neighbors*, we generate 50, match them but retrieve k instead of only the nearest neighbor and select the 50 most uncertain among all matched real neighbors. *Uncertain* refers to classical uncertainty sampling that we show as a reference. We conclude, that using ASAL to construct subsets and search for uncertain samples, leads to higher entropy of newly added samples and to a better classification accuracy on any subset size.

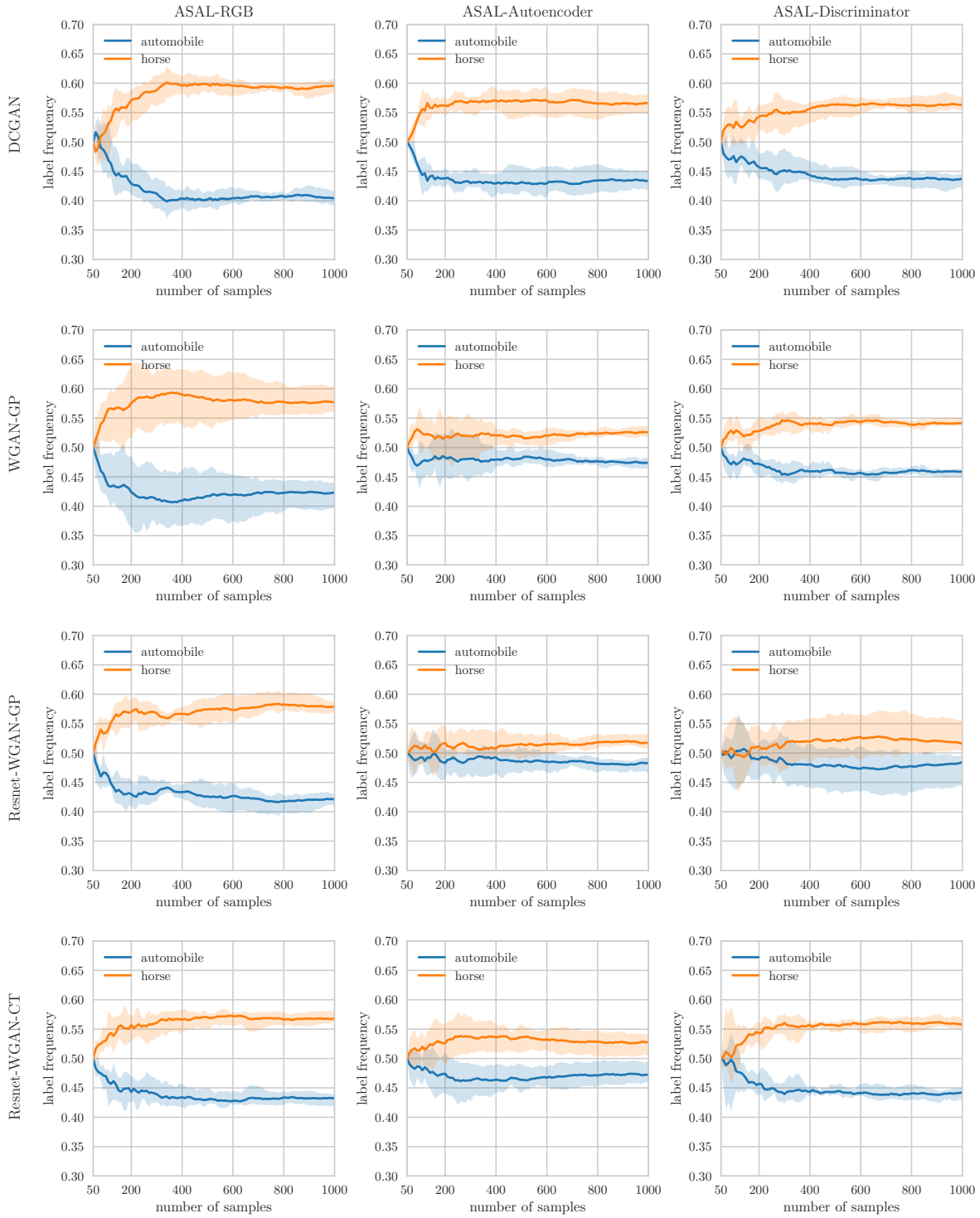


Figure 17: Label distribution for active learning with minimum distance sample generation and the Hinge loss, using different matching strategies and GANs for *CIFAR-10 - two classes*. All setups assemble training sets containing the more image with the label horse than automobile.

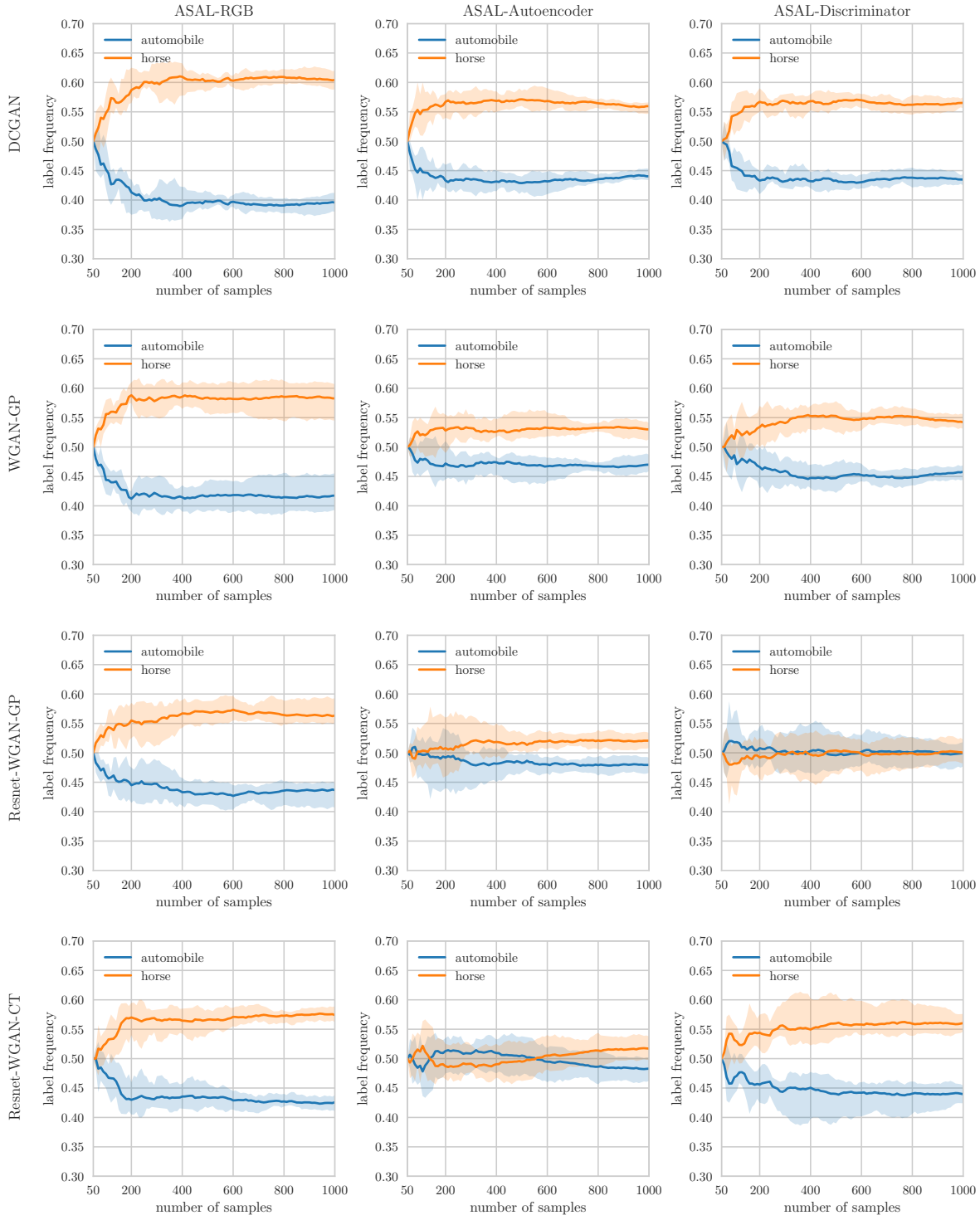


Figure 18: Label distribution for active learning with maximum entropy sample generation and the cross-entropy loss, using different matching strategies and GANs for *CIFAR-10 - two classes*.

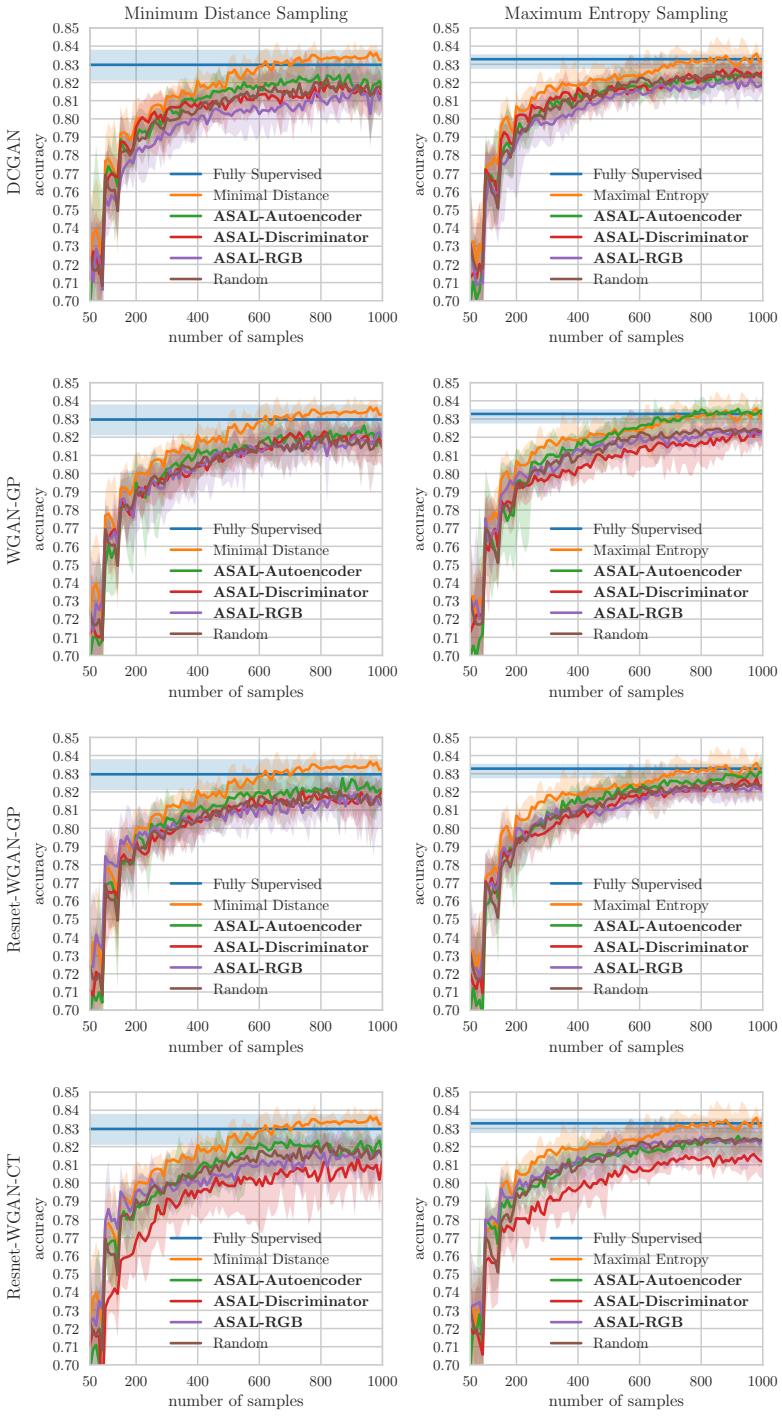


Figure 19: Validation accuracy on *CIFAR-10 - two classes* of a fully supervised model, for random sampling, uncertainty sampling and different ASALs using different GANs. ASAL-Autoencoder leads to the best performance. ASAL-Disc. using Resnet-WGAN-CT performs worse that any other strategy because the sample matching using is unable to retrieve high entropy samples from the pool, see Fig. 20.

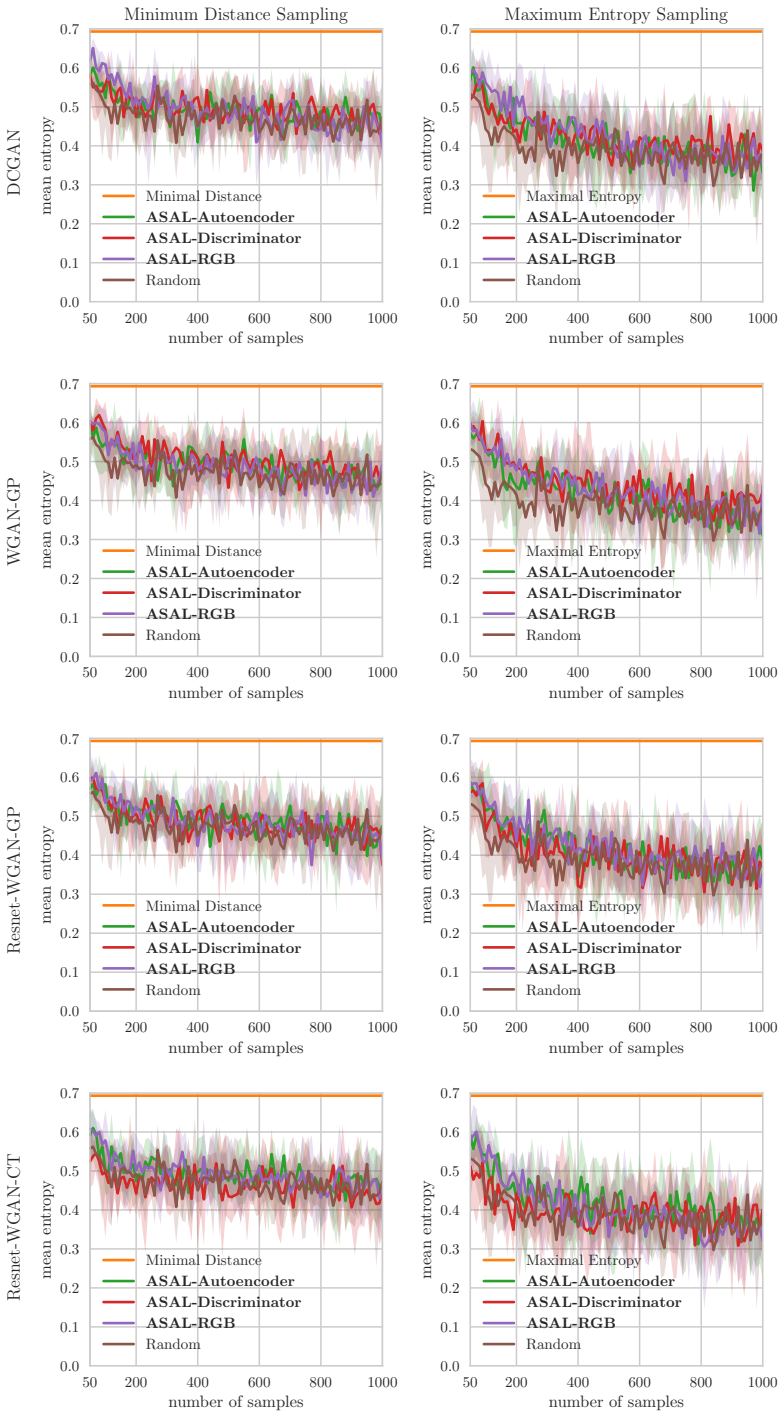
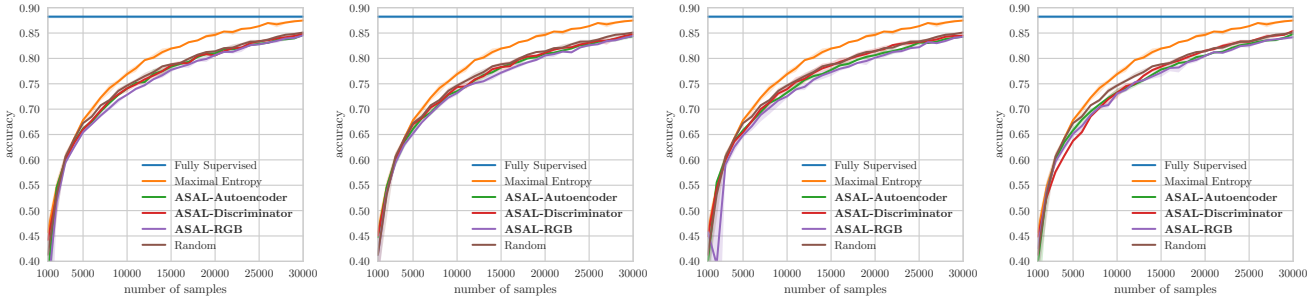
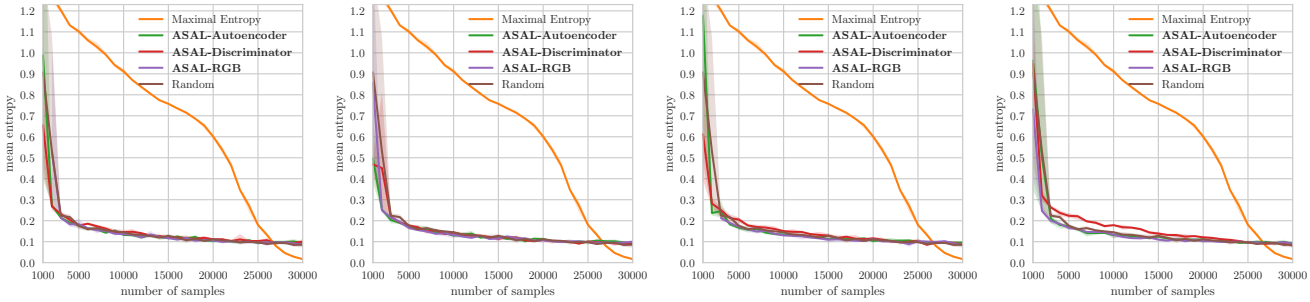


Figure 20: Average entropy of images that are selected and added to the training set for *CIFAR-10 - two classes* using different GANs. The mean entropy of the random sampling and the proposed method show hardly any difference. However, for maximum entropy sampling at least at the beginning ASAL selects images with higher entropy than random sampling.



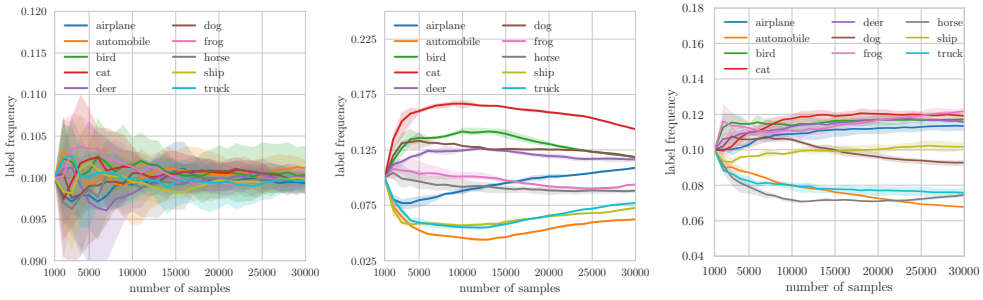
(a) DCGAN. (b) WGAN-GP. (c) Resnet-WGAN-GP. (d) Resnet-WGAN-CT.

Figure 21: Validation accuracy on *CIFAR-10 - ten classes* of a fully supervised model, for random sampling, uncertainty sampling and different ASALs using different GANs. The proposed method performs slightly worse than random sampling independent of the sample matching of GAN.



(a) DCGAN. (b) WGAN-GP. (c) Resnet-WGAN-GP. (d) Resnet-WGAN-CT.

Figure 22: Average entropy of images that are selected and added to the training set for *CIFAR-10 - ten classes* using different GANs. There is hardly any difference for random sampling and ASAL in the entropy of newly added samples. Only at the beginning, random sampling retrieves samples with slightly higher entropy.



(a) Random sampling. (b) *Maximum entropy* sampling. (c) *ASAL Classifer Features* sampling.

Figure 23: Label distribution for uncertainty sampling using maximum entropy and random sampling for *CIFAR-10 - ten classes*. Random sampling converges to the true label distribution in the pool. Maximum entropy sampling selects most frequently cat, dog, bird, deer and least frequently automobile, ship, truck to exceed the classification quality of random sampling.

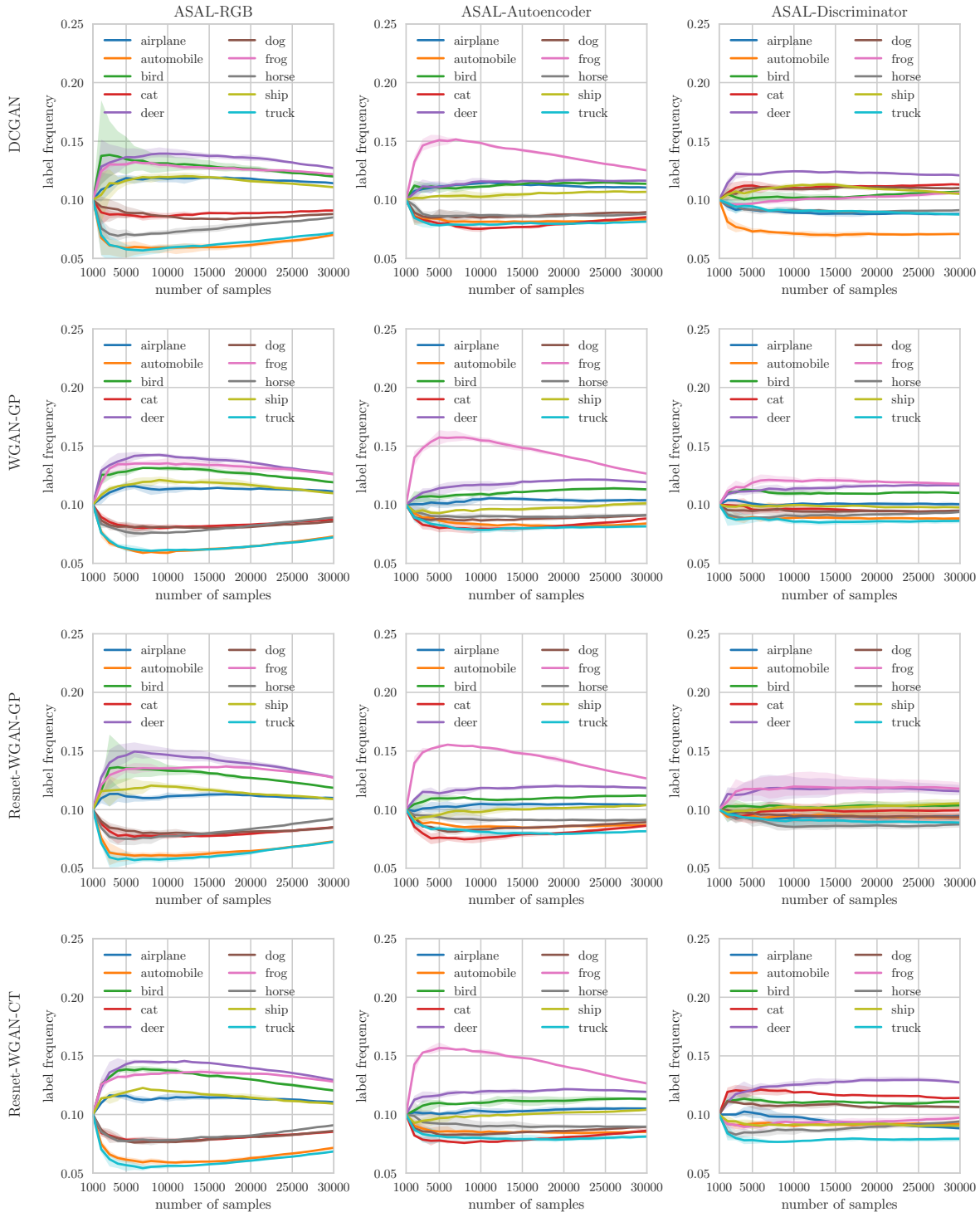


Figure 24: Label distribution for active learning using different matching strategies, uncertainty measures and GANs for *CIFAR-10 - ten classes*. Exactly the classes cat, dog that are most common in the training set of uncertainty sampling are less common in the data sets of most setups. Conversely, frog is for many setups the most common class but is not particularly frequent in the uncertainty sampling data set.

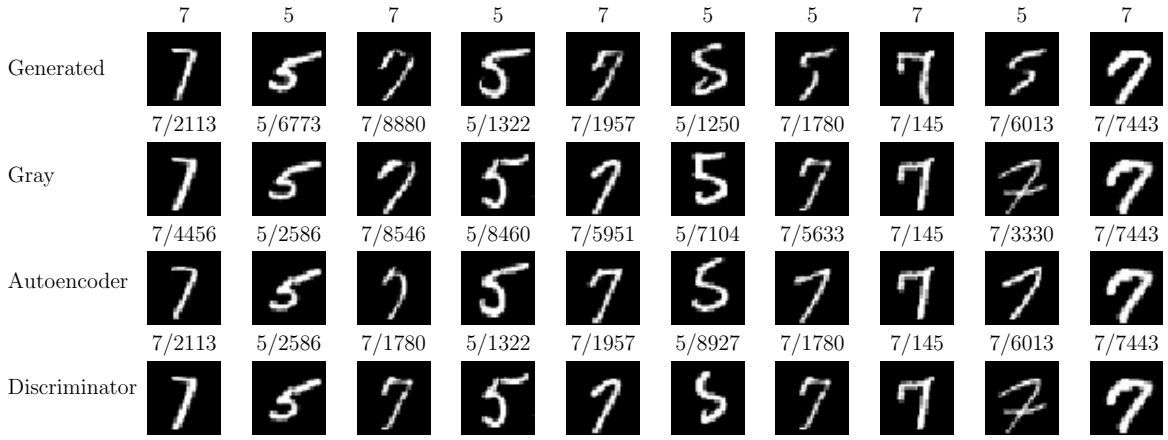


Figure 25: The first row shows synthetic digits and the other the closest samples from the pool using different features for comparison. The numbers above the image denote the label and image id.

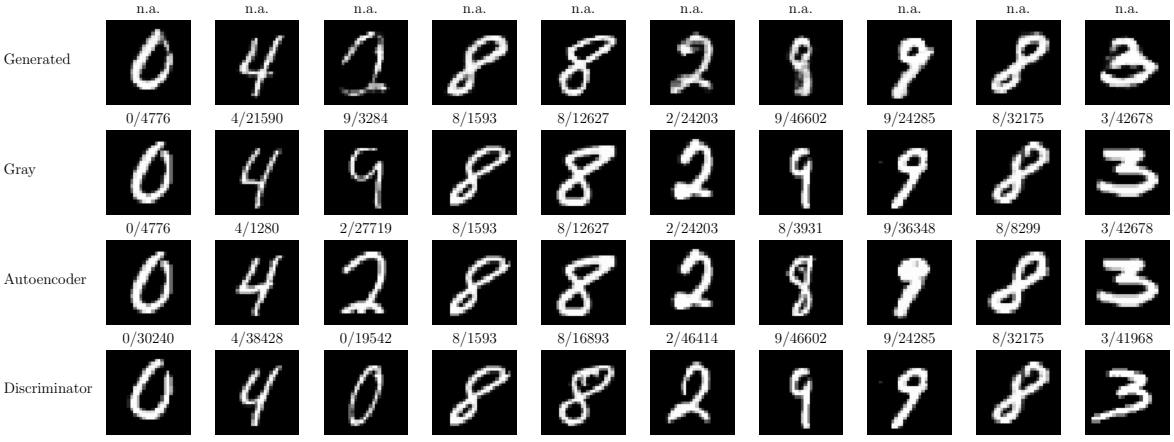


Figure 26: The rows show generated and matched images for *MNIST - ten classes* using WGAN-GP.

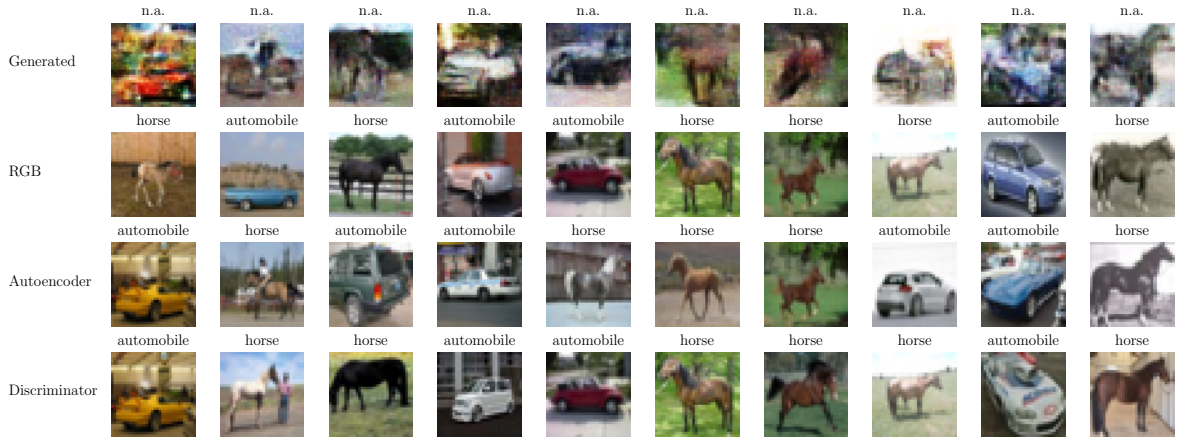


Figure 27: The rows show generated and matched images for *CIFAR-10 - two classes* using WGAN-GP. The images have a reasonable quality and all matching strategies retrieve images that are visually close or show the same class.

Figure 28: The rows show generated and matched images for *CIFAR-10 - ten classes* using WGAN-GP. Most of the generated images achieve only a moderate quality and even the closest samples from the pool have a high perceptual visual distance or assign images that show non matching classes, see last column where the images have a similar appearance but an appropriate label for the generated images would be horse but the selected samples show airplane and ship.