DISCO Nets: DISsimilarity COefficient Networks Supplementary material

Diane Bouchacourt
University of Oxford
diane@robots.ox.ac.uk

M. Pawan Kumar
University of Oxford
pawan@robots.ox.ac.uk

Sebastian Nowozin
Microsoft Research Cambridge
sebastian.nowozin@microsoft.com

In this supplementary material, we refer to equations and sections of the main paper, and to figures and tables of this supplementary material, unless otherwise stated.

1 Toy example experimental details.

In this section, we provide details on the toy example presented in Section 1. We used the following simple experimental setting. All covariances for the bidimensional distributions are diagonal, therefore all bidimensional Gaussian distributions are parametrised by 4 parameters $(\mu_1, \mu_2, \sigma_1, \sigma_2)$ where μ, σ is a mean-variance pair on each dimension. We consider a data distribution that is a mixture of 2 bidimensional Gaussian distributions, referred as $\mathcal{GMM}$. The first Gaussian of the mixture, $\mathcal{G}_1$, is parametrised by $(1, 1.5, 2, 0.8)$ and the second Gaussian $\mathcal{G}_2$ is parametrised by $(0, -0.5, 0.7, 0.6)$. The mixture weights are 0.7 and 0.3, such that $\mathcal{GMM} = 0.7 \times \mathcal{G}_1 + 0.3 \times \mathcal{G}_2$. We consider two models to capture the true data distribution $\mathcal{GMM}$. Each model is able to represent a bidimensional Gaussian distribution parametrised by $(\mu_1, \mu_2, \sigma_1, \sigma_2)$. The sets in which to search for the parameters are the same in both dimensions and both models. The set to search the means ranges from -3 to 3 by 1, and the set to search the variances ranges 0.1 to 2 by 0.5. The training dataset is composed of $N = 10000$ examples drawn randomly from $\mathcal{GMM}$, denoted as $(\boldsymbol{x}_1, ..., \boldsymbol{x}_N)$. The testing dataset is composed of 1000 examples drawn randomly from $\mathcal{GMM}$. During training, we draw $K = 2$ samples from the model and estimate the probabilistic loss defined as:

$$\frac{1}{N}\sum_{n=1}^{N}\Big[\frac{1}{K}\sum_{k}\Delta_M(\boldsymbol{x}_n, G_M(\boldsymbol{z}_k)) - \frac{1}{2}\frac{1}{K(K-1)}\sum_{k}\sum_{k'\neq k}\Delta_M(G_M(\boldsymbol{z}_{k'}), G_M(\boldsymbol{z}_k))\Big] \quad (1)$$

where M indexes the model, Δ_M is the model's specific loss, and $G(\boldsymbol{z}_k))_M$ is the k^{th} sample drawn from M for the training data $\boldsymbol{x}_n$. During testing of a model, we draw $K = 10$ samples from the model and choose a pointwise prediction with the MEU method. The MEU method employs the same loss as the evaluation loss. Each model is tested with its own loss and the loss of the other model.

2 Details on the MEU method

For an input $\boldsymbol{x}$, to choose a single prediction $\boldsymbol{y}$ among K candidate outputs sampled for $\boldsymbol{x}$, DISCO Nets use the principle of Maximum Expected Utility (MEU). The prediction $\boldsymbol{y}_{\Delta_{\text{task}}}$ maximises the expected utility, or rather minimises the expected task-specific loss Δ_{task}, estimated using the sampled candidates. Formally, the prediction is made as follows:

$$\boldsymbol{y}_{\Delta_{\text{task}}} = \underset{k\in[1,K]}{\operatorname{argmax}}\, \text{EU}(\boldsymbol{y}_k) = \underset{k\in[1,K]}{\operatorname{argmin}} \sum_{k'=1}^{K} \Delta_{\text{task}}(\boldsymbol{y}_k, \boldsymbol{y}'_k) \quad (2)$$

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

Table 1: Evaluation metrics. N is the number of testing example pairs $(\boldsymbol{x}_n, \boldsymbol{y}_n)$ and $\boldsymbol{y}_{\Delta_{\text{metric}},n}$ is the prediction, specific to the metric, for the n^{th} input example $\boldsymbol{x}_n$. J is the number of joints of the hand.

Shorthand & Definition	Δ_{metric}	Formula
ProbLoss: Probabilistic Loss of K sampled poses.	$\Delta_{\text{ProbLoss}} = \|\|.\|\|_2^\beta$	$\widehat{F}(\Delta_{\text{ProbLoss}}, \boldsymbol{\theta})$ with $\gamma = 0.5$
MeJEE (Mean Joint Euclidean Error) : per-joint Euclidean distance between the pointwise pose and the goundtruth, averaged by J and N.	$\Delta_{\text{MeJEE}} = \frac{1}{J} \sum_{j=1}^{J} \|\|.\|\|_2^{\text{joint}j}$	$\frac{1}{N} \sum_{n=1}^{N} \frac{1}{J} \sum_{j=1}^{J} \|\|\boldsymbol{y}_n^j - \boldsymbol{y}_{\Delta_{\text{MeJEE}},n}^j\|\|_2^{\text{joint}j}$
MaJEE (Max Joint Euclidean Error) per-joint maximal Euclidean distance between the pointwise pose and the goundtruth averaged by N.	$\Delta_{\text{MaJEE}} = \max_{j \in [1,J]} \|\|.\|\|_2^{\text{joint}j}$	$\frac{1}{N} \sum_{n=1}^{N} \max_{j \in [1,J]} \|\|\boldsymbol{y}_n^j - \boldsymbol{y}_{\Delta_{\text{MaJEE}},n}^j\|\|_2^{\text{joint}j}$
FF(d) (Fraction of Frames) : fraction of test examples that have all predicted joints of the pointwise pose below a given maximum Euclidean distance d in mm from the ground-truth.	$\Delta_{\text{FF}} = -\mathbb{1}_{\max_{j \in [1,J]} \|\|.\|\|_2^{\text{joint}j} \leq d}$ (note the minus sign since we want to maximise FF)	$\frac{1}{N} \sum_{n=1}^{N} \mathbb{1}_{\max_{j \in [1,J]} \|\|\boldsymbol{y}_n^j - \boldsymbol{y}_{\Delta_{\text{FF}},n}^j\|\|_2^{\text{joint}j} \leq d}$

where $(\boldsymbol{y_1}, ..., \boldsymbol{y}_K)$ are the candidates output corresponding to the single input $\boldsymbol{x}$. For example, for a given intput $\boldsymbol{x}$, we need to choose a pointwise output to evaluate the Mean Joint Euclidean Error (MeJEE). We sample K candidate ouputs values for the input $\boldsymbol{x}$, and pick:

$$\boldsymbol{y}_{\Delta_{\text{MeJEE}}} = \underset{k \in [1,K]}{\operatorname{argmin}} \sum_{k'=1}^{K} \Delta_{\text{task}}(\boldsymbol{y}_k, \boldsymbol{y}'_k) = \underset{k \in [1,K]}{\operatorname{argmin}} \sum_{k=1}^{K} \frac{1}{J} \sum_{j=1}^{J} ||\boldsymbol{y}^j - \boldsymbol{y}_k^j||_2 \tag{3}$$

Then when we evaluate MeJEE, the loss encountered on the example $\boldsymbol{x}$ is $\Delta_{\text{MeJEE}}(\boldsymbol{y}_{\text{GT}}, \boldsymbol{y}_{\Delta_{\text{MeJEE}}})$ where $\boldsymbol{y}_{\text{GT}}$ is the ground-truth output that corresponds to $\boldsymbol{x}$.

3 Experimental details

We provide in this section additional details on the Hand Pose experiment of Section 3. As in the main paper, we denote the training loss function $\Delta_{\text{training}} = \Delta_\beta(\boldsymbol{y}, \boldsymbol{y'}) = ||\boldsymbol{y} - \boldsymbol{y'}||_2^\beta$.

Evaluation metrics. Table 1 details the evaluation metrics we employ.

Cross-validation procedure. We substract $I = 10000$ examples from the 72757 training frames to construct a validation dataset. The validation examples are chosen at random and are the same for all experiments. Let us denote the examples pairs from the validation dataset as $V = \{\boldsymbol{x}_i, \boldsymbol{y}_i, i = 1..I\}$, and $\boldsymbol{y}_{\Delta_{\text{training}},i}$ is the prediction for the i^{th} example. During training, we monitor the value of the loss Δ_{training} on the validation dataset. In details, this loss is:

$$L_{\text{val}} = \frac{1}{I} \sum_{i=1}^{I} ||\boldsymbol{y}_i - \boldsymbol{y}_{\Delta_{\text{training}},i}||_2^\beta \tag{4}$$

In order to evaluate L_{val} for the model BASE$_\beta$, we simply use for $\boldsymbol{y}_{\Delta_{\text{training}}}$ the pointwise prediction of BASE$_\beta$. To monitor L_{val} for DISCO$_{\beta,\gamma}$, we use the MEU method to pick $\boldsymbol{y}_{\Delta_{\text{training}}}$. In the MEU method, we draw $K = 100$ samples per validation example. In order to reduce variance, we draw the 100 random noise vectors per example, $\{\boldsymbol{z}_{1,1}, ...\boldsymbol{z}_{i,K}, ..., \boldsymbol{z}_{I,1}, ...\boldsymbol{z}_{I,K}\}$ once for all before starting the optimisation Algorithm 1. Indeed, contrarily to the random noise vector drawn to estimate the gradient of the training objective in step 4 of Algorithm 1, these noise vectors do not influence the training but are only used for monitoring purposes. They remain independent when the network's parameters are optimised. Finally, we choose for each model the best value of C and the best seed by taking the setting that gives the lowest final value of L_{val}.

Training procedure All network weights are initialised at random with a Gaussian distribution of mean 0 and standard deviation 0.01, all biases are initialised to 0. During training, the number of candidates outputs generated by the probabilistic models DISCO$_{\beta,\gamma}$ is $K = 2$. This is sufficient to construct an unbiased estimate of the gradient in step 7 of Algorithm 1 in the main paper. Note that in step 4 of Algorithm 1 we must draw new noise samples $(\boldsymbol{z}_{n,1}, ...\boldsymbol{z}_{n,k})$ for each training example at each iteration. Indeed, let us consider the iteration t on a training example $\boldsymbol{x}_n$. The values of the noise sampled for $\boldsymbol{x}_n$ at iteration $t-1$, $(\boldsymbol{z}_{n,1}^{t-1}, ...\boldsymbol{z}_{n,k}^{t-1})$, were used in the estimation of the gradient, and thus influenced the update $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1}$. Thus, we need to draw new sample to ensure that the K candidate outputs for $\boldsymbol{x}_n$, $\boldsymbol{y}_n = G(\boldsymbol{z}_{n,k}^t, \boldsymbol{x}_n; \boldsymbol{\theta}_t), k = 1..K$, remain independent given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}_t$. We train all models for 400 epochs as it results in a change of less than 3% in the value of L_{val} for BASE$_\beta$. Convergence behavior is shown in Figures 1, 2 and 3.

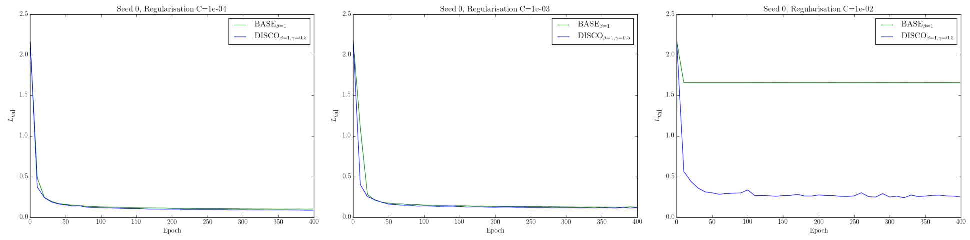


Figure 1: L_{val} monitoring for the two models, for different values of C, for seed 0.

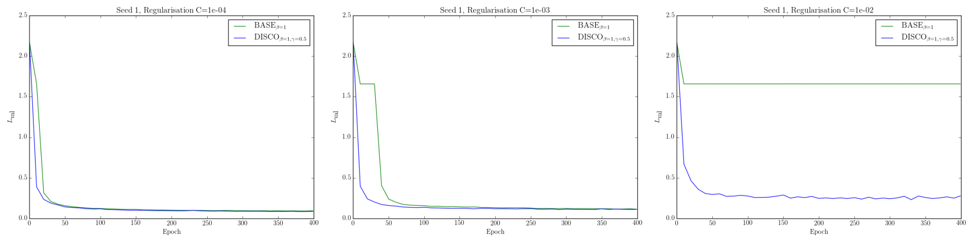


Figure 2: L_{val} monitoring for the two models, for different values of C, for seed 1.

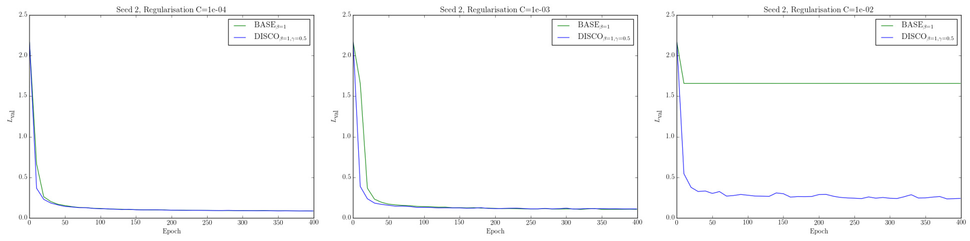


Figure 3: L_{val} monitoring for the two models, for different values of C, for seed 2.

Details on the qualitative results. We estimate the cumulative distribution function of the mean euclidean metric (MeJEE) using $K = 1000$ sampled poses for a given test image. This distribution is better fitted (lower BIC score, using 1000 sampled poses per image) by a bimodal Gaussian than a unimodal Gaussian for 3% of the test set (247 out of the 8252 images). It is never the case when $\gamma = 0$ if we do not scale the noise vector (when scaled, the distribution is multimodal for 217 test images). It shows the ability of DISCO Net to capture multimodality.

Details on the quantitative results. We detail here some results of the experiment presented in Section for all values of C, for the best seed for each C value. We report the BASE$_{\beta=1,\sigma}$ and DISCO$_{\beta=1,\gamma=0.5}$ models. In the main paper we report performances using the MEU method for choosing a pointwise prediction $\boldsymbol{y}_\triangle$ for an input $\boldsymbol{x}$. However we can use 3 different methods that are:

- MEU method.
- MEAN method : pick for the non-probabilistic model BASE$_{\beta,\sigma}$ its pointwise prediction. Therefore the value of the metrics MeJEE, MaJEE and FF are the same regardless of σ. For DISCO$_{\gamma,\beta}$, pick the mean of the K candidates.

- RANDOM method : pick an output $\boldsymbol{y}$ at random among K candidates.

Detailed results show that the model DISCO$_{\beta=1,\gamma=0.5}$ consistently outperforms BASE$_{\beta=1,\sigma}$ for all values of C and all methods, and that results are consistent across the pointwise prediction method employed. Tables 2, 3 and 4 present the results when we use the MEU method to choose the pointwise estimate.Tables 5, 6 and 7 present the results when we use the MEAN method. Tables 8, 9 and 10 present the results when we use the RANDOM method.

Table 2: Metrics values ± SEM for $C = 1e^{-2}$ using MEU.

Model	ProbLoss	MeJEE (mm)	MaJEE (mm)	FF (80mm)
BASE$_{\beta=1,\sigma=1}$	210.1±0.793	51.9±0.202	92.5±0.325	38.451
BASE$_{\beta=1,\sigma=5}$	204.8±0.792	52.0±0.201	92.7±0.325	38.257
BASE$_{\beta=1,\sigma=10}$	201.1±0.791	52.2±0.201	92.8±0.324	37.215
DISCO$_{\beta=1,\gamma=0.5}$	87.8±0.506	23.5±0.129	50.9±0.264	88.900

Table 3: Metrics values ± SEM for $C = 1e^{-3}$ using MEU.

Model	ProbLoss	MeJEE (mm)	MaJEE (mm)	FF (80mm)
BASE$_{\beta=1,\sigma=1}$	100.8±0.586	24.5±0.142	51.0±0.271	88.742
BASE$_{\beta=1,\sigma=5}$	96.3±0.579	24.8±0.141	51.2±0.270	88.548
BASE$_{\beta=1,\sigma=10}$	93.3±0.571	25.1±0.140	51.5±0.268	88.488
DISCO$_{\beta=1,\gamma=0.5}$	80.4±0.490	20.6±0.123	44.5±0.245	94.292

Table 4: Metrics values ± SEM for $C = 1e^{-4}$ using MEU.

Model	ProbLoss	MeJEE (mm)	MaJEE (mm)	FF (80mm)
BASE$_{\beta=1,\sigma=1}$	103.8±0.627	25.2±0.152	52.7±0.290	86.040
BASE$_{\beta=1,\sigma=5}$	99.3±0.620	25.5±0.151	52.9±0.289	85.773
BASE$_{\beta=1,\sigma=10}$	96.3±0.612	25.7±0.149	53.2±0.288	85.664
DISCO$_{\beta=1,\gamma=0.5}$	83.8±0.503	20.9±0.124	45.1±0.246	94.438

Table 5: Metrics values ± SEM for $C = 1e^{-2}$ using MEAN.

Model	ProbLoss	MeJEE (mm)	MaJEE (mm)	FF (80mm)
BASE$_{\beta=1,\sigma=1}$	210.1±0.793	51.8±0.202	92.5±0.325	38.536
BASE$_{\beta=1,\sigma=5}$	204.8±0.792	51.8±0.202	92.5±0.325	38.536
BASE$_{\beta=1,\sigma=10}$	201.1±0.791	51.8±0.202	92.5±0.325	38.536
DISCO$_{\beta=1,\gamma=0.5}$	87.8±0.506	23.5±0.129	50.7±0.263	90.221

Table 6: Metrics values ± SEM for $C = 1e^{-3}$ using MEAN.

Model	ProbLoss	MeJEE (mm)	MaJEE (mm)	FF (80mm)
BASE$_{\beta=1,\sigma=1}$	100.8±0.586	24.4±0.142	50.9±0.271	88.657
BASE$_{\beta=1,\sigma=5}$	96.3±0.579	24.4±0.142	50.9±0.271	88.657
BASE$_{\beta=1,\sigma=10}$	93.3±0.571	24.4±0.142	50.9±0.271	88.657
DISCO$_{\beta=1,\gamma=0.5}$	80.4±0.490	20.6±0.123	44.3±0.244	94.535

Table 7: Metrics values ± SEM for $C = 1e^{-4}$ using MEAN.

Model	ProbLoss	MeJEE (mm)	MaJEE (mm)	FF (80mm)
BASE$_{\beta=1,\sigma=1}$	103.8±0.627	25.1±0.152	52.7±0.290	85.991
BASE$_{\beta=1,\sigma=5}$	99.3±0.620	25.1±0.152	52.7±0.290	85.991
BASE$_{\beta=1,\sigma=10}$	96.3±0.612	25.1±0.152	52.7±0.290	85.991
DISCO$_{\beta=1,\gamma=0.5}$	83.8±0.503	20.9±0.124	45.0±0.246	94.619

Table 8: Metrics values $\pm$ SEM for $C = 1e^{-2}$ using RANDOM.

Model	ProbLoss	MeJEE (mm)	MaJEE (mm)	FF (80mm)
BASE$_{\beta=1,\sigma=1}$	210.1$\pm$0.793	51.9$\pm$0.202	92.5$\pm$0.325	38.500
BASE$_{\beta=1,\sigma=5}$	204.8$\pm$0.792	52.1$\pm$0.201	92.8$\pm$0.324	37.700
BASE$_{\beta=1,\sigma=10}$	201.1$\pm$0.791	52.4$\pm$0.201	93.1$\pm$0.325	37.082
DISCO$_{\beta=1,\gamma=0.5}$	87.8$\pm$0.506	24.1$\pm$0.130	52.1$\pm$0.266	88.572

Table 9: Metrics values $\pm$ SEM for $C = 1e^{-3}$ using RANDOM.

Model	ProbLoss	MeJEE (mm)	MaJEE (mm)	FF (80mm)
BASE$_{\beta=1,\sigma=1}$	100.8$\pm$0.586	24.6$\pm$0.142	51.0$\pm$0.271	88.694
BASE$_{\beta=1,\sigma=5}$	96.3$\pm$0.579	25.0$\pm$0.140	51.4$\pm$0.269	88.500
BASE$_{\beta=1,\sigma=10}$	93.3$\pm$0.571	25.5$\pm$0.138	51.8$\pm$0.267	88.585
DISCO$_{\beta=1,\gamma=0.5}$	80.4$\pm$0.490	20.9$\pm$0.123	45.0$\pm$0.246	94.062

Table 10: Metrics values $\pm$ SEM for$C = 1e^{-4}$ using RANDOM.

Model	ProbLoss	MeJEE (mm)	MaJEE (mm)	FF (80mm)
BASE$_{\beta=1,\sigma=1}$	103.8$\pm$0.627	25.3$\pm$0.151	52.8$\pm$0.290	86.028
BASE$_{\beta=1,\sigma=5}$	99.3$\pm$0.620	25.7$\pm$0.150	53.1$\pm$0.288	85.761
BASE$_{\beta=1,\sigma=10}$	96.3$\pm$0.612	26.1$\pm$0.148	53.5$\pm$0.286	85.822
DISCO$_{\beta=1,\gamma=0.5}$	83.8$\pm$0.503	21.0$\pm$0.124	45.3$\pm$0.246	94.510

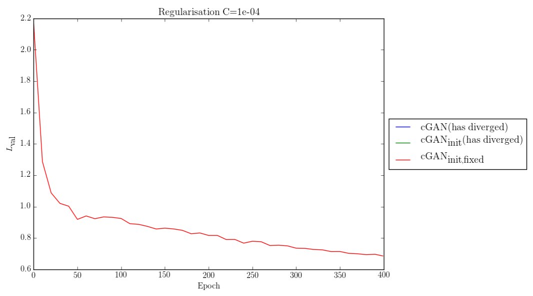
Detailed comparison with cGAN. We aim at performing a fair comparison with the conditional Generative Adversarial Networks (cGAN) model presented in Mirza and Osindero [1]. However as mentioned in Section 4.3 we encountered several challenges. We present here additional details on the comparison with cGAN models.

As mentioned in Section 4.3, we tried 3 different training setting in order to apply cGAN to hand pose estimation on the NYU dataset. The setting cGAN initialises randomly the Discriminator and the Generator parameters. The setting cGAN$_{\text{init}}$ initialises the convolutional layers of both the Discriminator and the Generator with the trained convolutional layers of our best DISCO Nets from Section 3 without keeping it fixed. That is, we try to perform fine-tuning only. The setting cGAN$_{\text{init, fixed}}$ initialises the convolutional layers of both the Discriminator and the Generator with the trained convolutional layers of our best DISCO Nets from Section 3 and keep these part fixed. That is, the convolutional parts of the Generator and the Discriminator are feature extractors that are not trained. This is a setting similar to the one employed for tag-annotation of images in Mirza and Osindero [1].

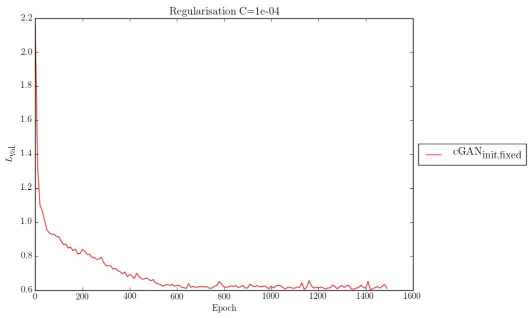
Since cGAN model require the training of the additional Discriminator network, each epoch of training requires ~ 60 seconds compared to ~ 20 seconds for DISCO Nets. Therefore, we were only able to use one random seed for the initialisation of the network parameters. However, DISCO Nets present a consistent behavior regardless of the seed employed for initialisation. We could expect a similar behavior for the cGAN model. The experimental setting is similar to the one of Section 3. We use the exact same training and validation sets as in Section 3. Back-propagation was used with Stochastic Gradient Descent. The learning rate is fixed to $\lambda = 0.01$ and we use a momentum of $m = 0.9$. We use a batchsize of 256 samples. We also add L2-regularisation controlled by a parameter C. We use $C = [1e^{-4}, 1e{-}3, 1e{-}2]$. We use the cross-validation procedure presented in Section 3 of this supplementary to cross-validate C. We train all models for 400 epochs. For $C = 1e^{-4}$ we train for more epochs since 400 were not enough to have the final convergence of cGAN$_{\text{init, fixed}}$ (see Figure 4b). However, this does not help the cGAN performances compared to the one reported for 400 epochs, see Table 11. Figure 4 show the training behavior of the different settings of cGAN. When the curve is missing, it means that the model has diverged after the few first iterations and thus we cannot show its behavior. This is always the case forcGAN$_{\text{init}}$.

Table 11: Metrics values $\pm$ SEM for$C = 1e^{-4}$ for cGAN model.

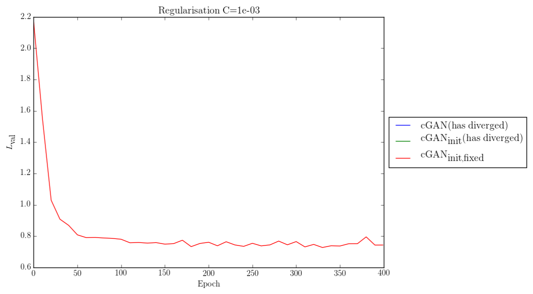
Model	ProbLoss (mm)	MeJEE (mm)	MaJEE (mm)	FF (80mm)
$\text{cGAN}_{\text{init, fixed}}$, 400 epochs of training	128.9$\pm$0.480	31.8$\pm$0.117	64.3$\pm$0.230	78.454
$\text{cGAN}_{\text{init, fixed}}$, >1400 epochs of training	134.5$\pm$0.591	31.8$\pm$0.146	64.0$\pm$0.253	74.261



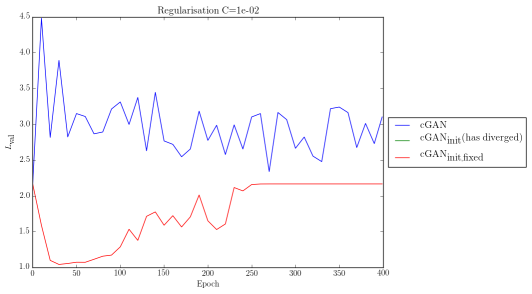
(a) L_{val} monitoring for cGan with $C = 1e^{-4}$ for 400 epochs.



(b) L_{val} monitoring for cGan with $C = 1e^{-4}$. We trained for more than 1400 epochs as 400 epochs were not enough.



(c) L_{val} monitoring for cGan with $C = 1e^{-3}$.



(d) L_{val} monitoring for cGan with $C = 1e^{-2}$.

Figure 4: L_{val} monitoring for cGan.

References

[1] M. Mirza and S. Osindero. Conditional generative adversarial nets. In *NIPS Deep Learning Workshop*, 2014.

[2] Pierre Pinson and Julija Tastu. Discrimination ability of the energy score. *Technical University of Denmark. (DTU Compute-Technical Report-2013; No. 15)*, 2013.