Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \mathcal{W}(\mathcal{G}_\theta, \mu)$$

- $\mathcal{W}$ is the Wasserstein 1-distance, quantifies similarity of probability measures.
- Generator: $\mathcal{G}_\theta$ is a probability distribution on X.
- Desired property: $\mathcal{G}_{\hat{\theta}} \approx \mu$.

Issue: μ unknown $\implies$ not possible to evaluate $\mathcal{W}(\mathcal{G}_\theta, \mu)$.

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\mathrm{D} \in \mathrm{Lip}(X)} \left\{ \mathbb{E}_{\mathbf{x} \sim \mu}\big[\mathrm{D}(\mathbf{x})\big] - \mathbb{E}_{\mathbf{v} \sim \mathcal{G}_\theta}\big[\mathrm{D}(\mathbf{v})\big] \right\} \right\}$$

- Discriminator: $\mathrm{D} \colon X \to \mathbb{R}$ is a 1-Lipschitz function.
- Generator: $\mathcal{G}_\theta$ is a probability distribution on X.
- Desired property: $\mathcal{G}_{\hat{\theta}} \approx \mu$.

Issue: μ unknown $\implies$ not possible to evaluate $\mathcal{W}\big(\mathcal{G}_\theta, \mu\big)$.
$\implies$ Re-write using the Kantorovich-Rubinstein dual characterization of $\mathcal{W}$.

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\mathrm{D} \in \mathrm{Lip}(X)} \left\{ \mathbb{E}_{\mathbf{x} \sim \mu} \left[\mathrm{D}(\mathbf{x}) \right] - \mathbb{E}_{\mathbf{v} \sim \mathcal{G}_\theta} \left[\mathrm{D}(\mathbf{v}) \right] \right\} \right\}$$

- Discriminator: $\mathrm{D} \colon X \to \mathbb{R}$ is a 1-Lipschitz function.
- Generator: $\mathcal{G}_\theta$ is a probability distribution on X.
- Desired property: $\mathcal{G}_{\hat{\theta}} \approx \mu$.

Issue: How to ensure D is 1-Lipschitz?

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\mathrm{D}\colon X \to \mathbb{R}} \left\{ \mathbb{E}_{\mathbf{x}\sim\mu}\left[\mathrm{D}(\mathbf{x})\right] - \mathbb{E}_{\mathbf{v}\sim\mathcal{G}_\theta}\left[\mathrm{D}(\mathbf{v})\right] + \lambda\, \mathbb{E}_{\tilde{\mathbf{x}}\sim\tilde{\mu}}\left[\left(\left\| \partial\, \mathrm{D}(\tilde{\mathbf{x}}) \right\| - 1 \right)^2 \right] \right\} \right\}$$

- Discriminator: $\mathrm{D}\colon X \to \mathbb{R}$ is a measurable function.
- Generator: $\mathcal{G}_\theta$ is a probability distribution on X.
- Desired property: $\mathcal{G}_{\hat{\theta}} \approx \mu$.
- $\tilde{\mu}$ probability measure generated by $\boldsymbol{\epsilon}\mathbf{x} + (1 - \boldsymbol{\epsilon})\mathbf{v}$ with $\boldsymbol{\epsilon} \sim U(0, 1)$.

Issue: How to ensure D is 1-Lipschitz?
$\Longrightarrow$ softly enforce this condition by adding a penalty term.

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\mathrm{D}\colon X \to \mathbb{R}} \left\{ \mathbb{E}_{\mathbf{x} \sim \mu} \left[\mathrm{D}(\mathbf{x}) \right] - \mathbb{E}_{\mathbf{v} \sim \mathcal{G}_\theta} \left[\mathrm{D}(\mathbf{v}) \right] + \lambda \, \mathbb{E}_{\tilde{\mathbf{x}} \sim \tilde{\mu}} \left[\left(\left\| \partial \, \mathrm{D}(\tilde{\mathbf{x}}) \right\| - 1 \right)^2 \right] \right\} \right\}$$

- Discriminator: $\mathrm{D}\colon X \to \mathbb{R}$ is a measurable function.
- Generator: $\mathcal{G}_\theta$ is a probability distribution on X.
- Desired property: $\mathcal{G}_{\hat{\theta}} \approx \mu$.
- $\tilde{\mu}$ probability measure generated by $\boldsymbol{\epsilon}\mathbf{x} + (1 - \boldsymbol{\epsilon})\mathbf{v}$ with $\boldsymbol{\epsilon} \sim U(0, 1)$.

Issue: Unfeasible to maximise over all measurable mappings

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\phi} \left\{ \mathbb{E}_{\mathbf{x}\sim\mu}\left[\mathrm{D}_{\phi}(\mathbf{x}) \right] - \mathbb{E}_{\mathbf{v}\sim\mathcal{G}_{\theta}}\left[\mathrm{D}_{\phi}(\mathbf{v}) \right] + \lambda\, \mathbb{E}_{\tilde{\mathbf{x}}\sim\tilde{\mu}}\left[\left(\left\| \partial\, \mathrm{D}_{\phi}(\tilde{\mathbf{x}}) \right\| - 1 \right)^2 \right] \right\} \right\}$$

- Discriminator: $\mathrm{D}_{\phi} \colon X \to \mathbb{R}$ a function parametrised by neural network.
- Generator: $\mathcal{G}_{\theta}$ is a probability distribution on X.
- Desired property: $\mathcal{G}_{\hat{\theta}} \approx \mu$.
- $\tilde{\mu}$ probability measure generated by $\boldsymbol{\epsilon}\mathbf{x} + (1-\boldsymbol{\epsilon})\mathbf{v}$ with $\boldsymbol{\epsilon} \sim U(0,1)$.

Issue: Unfeasible to maximise over all measurable mappings
$\implies$ Parametrise discriminator by a neural network.

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\phi} \left\{ \mathbb{E}_{\mathbf{x}\sim\mu}\left[\mathrm{D}_{\phi}(\mathbf{x})\right] - \mathbb{E}_{\mathbf{v}\sim\mathcal{G}_{\theta}}\left[\mathrm{D}_{\phi}(\mathbf{v})\right] + \lambda\,\mathbb{E}_{\tilde{\mathbf{x}}\sim\tilde{\mu}}\left[\left(\left\|\partial\,\mathrm{D}_{\phi}(\tilde{\mathbf{x}})\right\| - 1\right)^2\right] \right\} \right\}$$

- Discriminator: $\mathrm{D}_{\phi} \colon X \to \mathbb{R}$ a function parametrised by neural network.
- Generator: $\mathcal{G}_{\theta}$ is a probability distribution on X.
- Desired property: $\mathcal{G}_{\hat{\theta}} \approx \mu$.
- $\tilde{\mu}$ probability measure generated by $\boldsymbol{\epsilon}\mathbf{x} + (1-\boldsymbol{\epsilon})\mathbf{v}$ with $\boldsymbol{\epsilon} \sim U(0,1)$.

Issue: Difficult to parametrise family of probability distributions on X

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\phi} \left\{ \mathbb{E}_{\mathbf{x}\sim\mu}\left[\mathrm{D}_\phi(\mathbf{x})\right] - \mathbb{E}_{\mathbf{z}\sim\sigma}\left[\mathrm{D}_\phi\left(\mathrm{G}_\theta(\mathbf{z})\right)\right] + \lambda\, \mathbb{E}_{\tilde{\mathbf{x}}\sim\tilde{\mu}}\left[\left(\left\|\partial\, \mathrm{D}_\phi(\tilde{\mathbf{x}})\right\| - 1\right)^2\right] \right\} \right\}$$

- Discriminator: $\mathrm{D}_\phi \colon X \to \mathbb{R}$ a function parametrised by neural network.
- Generator: $\mathrm{G}_\theta \colon Z \to X$ a function parametrised by neural network.
- Desired property: $\mathrm{G}_{\hat{\theta}}(\mathbf{z})$ is approximately μ-distributed when $\mathbf{z} \sim \sigma$ (σ known).
- $\tilde{\mu}$ probability measure generated by $\boldsymbol{\epsilon}\mathbf{x} + (1-\boldsymbol{\epsilon})\,\mathrm{G}_\theta(\mathbf{z})$ with $\boldsymbol{\epsilon} \sim U(0,1)$.

Issue: Difficult to parametrise family of probability distributions on X
$\implies$ Write generator as deterministic function with random input $\mathbf{z} \sim \sigma$ with σ known.

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\phi} \left\{ \mathbb{E}_{\mathbf{x} \sim \mu} \left[\mathrm{D}_{\phi}(\mathbf{x}) \right] - \mathbb{E}_{\mathbf{z} \sim \sigma} \left[\mathrm{D}_{\phi}\left(\mathrm{G}_{\theta}(\mathbf{z}) \right) \right] + \lambda \, \mathbb{E}_{\tilde{\mathbf{x}} \sim \tilde{\mu}} \left[\left(\left\| \partial \, \mathrm{D}_{\phi}(\tilde{\mathbf{x}}) \right\| - 1 \right)^2 \right] \right\} \right\}$$

- Discriminator: $\mathrm{D}_{\phi} \colon X \to \mathbb{R}$ a function parametrised by neural network.
- Generator: $\mathrm{G}_{\theta} \colon Z \to X$ a function parametrised by neural network.
- Desired property: $\mathrm{G}_{\hat{\theta}}(\mathbf{z})$ is approximately μ-distributed when $\mathbf{z} \sim \sigma$ (σ known).
- $\tilde{\mu}$ probability measure generated by $\boldsymbol{\epsilon}\mathbf{x} + (1 - \boldsymbol{\epsilon})\, \mathrm{G}_{\theta}(\mathbf{z})$ with $\boldsymbol{\epsilon} \sim U(0, 1)$.

Issue: μ unknown $\implies$ not possible to compute μ- and $\tilde{\mu}$-expectations

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\phi} \left\{ \frac{1}{m} \sum_{i=1}^{m} \left[\mathrm{D}_\phi(x_i) \right] - \mathbb{E}_{\mathbf{z}\sim\sigma} \left[\mathrm{D}_\phi\big(\mathrm{G}_\theta(\mathbf{z})\big) \right] + \lambda \frac{1}{m} \sum_{i=1}^{m} \left[\Big(\big\| \partial\, \mathrm{D}_\phi(\tilde{x}_i) \big\| - 1 \Big)^2 \right] \right\} \right\}$$

- Discriminator: $\mathrm{D}_\phi \colon X \to \mathbb{R}$ a function parametrised by neural network.
- Generator: $\mathrm{G}_\theta \colon Z \to X$ a function parametrised by neural network.
- Desired property: $\mathrm{G}_{\hat{\theta}}(\mathbf{z})$ is approximately μ-distributed when $\mathbf{z} \sim \sigma$ (σ known).
- $\tilde{\mu}$ probability measure generated by $\boldsymbol{\epsilon}\mathbf{x} + (1 - \boldsymbol{\epsilon})\, \mathrm{G}_\theta(\mathbf{z})$ with $\boldsymbol{\epsilon} \sim U(0,1)$.

Issue: μ unknown $\implies$ not possible to compute μ- and $\tilde{\mu}$-expectations
$\implies$ Use empirical distributions given by x_i and $\tilde{x}_i = \epsilon_i x_i + (1 - \epsilon_i)\, \mathrm{G}_\theta(z_i)$.

Wasserstein GAN

Input: Unsupervised data $x_i \in X$ generated by $\mathbf{x} \sim \mu$.
Task: Sample from $\mathbf{x} \sim \mu$ where μ is unknown.
Method: Learn how to sample from μ by solving (Arjovsky et al., 2017)

$$\hat{\theta} \in \arg\min_{\theta} \Big\{ \max_{\phi} \Big\{ \frac{1}{m} \sum_{i=1}^{m} \big[\mathrm{D}_\phi(x_i) \big] - \mathbb{E}_{\mathbf{z}\sim\sigma} \big[\mathrm{D}_\phi(\mathrm{G}_\theta(\mathbf{z})) \big] + \lambda \frac{1}{m} \sum_{i=1}^{m} \Big[\Big(\big\| \partial\, \mathrm{D}_\phi(\tilde{x}_i) \big\| - 1 \Big)^2 \Big] \Big\} \Big\}$$

- Discriminator: $\mathrm{D}_\phi \colon X \to \mathbb{R}$ a function parametrised by neural network.
- Generator: $\mathrm{G}_\theta \colon Z \to X$ a function parametrised by neural network.
- Desired property: $\mathrm{G}_{\hat{\theta}}(\mathbf{z})$ is approximately μ-distributed when $\mathbf{z} \sim \sigma$ (σ known).
- $\tilde{\mu}$ probability measure generated by $\boldsymbol{\epsilon}\mathbf{x} + (1 - \boldsymbol{\epsilon})\, \mathrm{G}_\theta(\mathbf{z})$ with $\boldsymbol{\epsilon} \sim U(0, 1)$.

Conclusion: Above expression is suitable for deep learning.

Conditional Wasserstein GAN

Input: Supervised data $(x_i, y_i) \in X \times Y$ generated by $(\mathbf{x}, \mathbf{y}) \sim \mu$.
Task: Sample from $(\mathbf{x} \mid \mathbf{y} = y) \sim \pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$ where posterior is unknown.
Method: Learn how to sample from posterior by solving

$$\hat{\theta} \in \arg\min_{\theta} \mathbb{E}_{\mathbf{y}} \Big[\mathcal{W}\big(\mathcal{G}_\theta(\mathbf{y}), \pi_{\text{post}}(\mathbf{x} \mid \mathbf{y})\big) \Big]$$

- $\mathcal{W}$ quantifies similarity between $\mathcal{G}_\theta(y)$ and $\pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$.
- Generator: $\mathcal{G}_\theta(y)$ is a probability distribution on X.
- Desired property: $\mathcal{G}_{\hat{\theta}}(y) \approx \pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$.

Conditional Wasserstein GAN

Input: Supervised data $(x_i, y_i) \in X \times Y$ generated by $(\mathbf{x}, \mathbf{y}) \sim \mu$.
Task: Sample from $(\mathbf{x} \mid \mathbf{y} = y) \sim \pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$ where posterior is unknown.
Method: Learn how to sample from posterior by solving

$$\hat{\theta} \in \arg\min_{\theta} \mathbb{E}_{\mathbf{y}} \Big[\mathcal{W}\big(\mathcal{G}_\theta(\mathbf{y}), \pi_{\text{post}}(\mathbf{x} \mid \mathbf{y})\big) \Big]$$

- $\mathcal{W}$ quantifies similarity between $\mathcal{G}_\theta(y)$ and $\pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$.
- Generator: $\mathcal{G}_\theta(y)$ is a probability distribution on X.
- Desired property: $\mathcal{G}_{\hat{\theta}}(y) \approx \pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$.

Repeating the same procedure as for Wasserstein GAN ...

Conditional Wasserstein GAN

Input: Supervised data $(x_i, y_i) \in X \times Y$ generated by $(\mathbf{x}, \mathbf{y}) \sim \mu$.
Task: Sample from $(\mathbf{x} \mid \mathbf{y} = y) \sim \pi_{\mathrm{post}}(\mathbf{x} \mid \mathbf{y} = y)$ where posterior is unknown.
Method: Learn how to sample from posterior by solving

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\phi} \left\{ \frac{1}{m} \sum_{i=1}^{m} \left[\mathrm{D}_{\phi}(x_i, y_i) - \mathbb{E}_{\mathbf{z} \sim \sigma} \left[\mathrm{D}_{\phi}\left(\mathrm{G}_{\theta}(\mathbf{z}, y_i), y_i \right) \right] \right] \right\} \right\}$$

- Discriminator: $\mathrm{D}_{\phi} \colon X \times Y \to \mathbb{R}$.
- Generator: $\mathrm{G}_{\theta} \colon Z \times Y \to X$.
- Desired property: $\mathrm{G}_{\theta}(\mathbf{z}, y)$ approximately $\pi_{\mathrm{post}}(\mathbf{x} \mid \mathbf{y} = y)$-distributed when $\mathbf{z} \sim \sigma$.

Conditional Wasserstein GAN

Input: Supervised data $(x_i, y_i) \in X \times Y$ generated by $(\mathbf{x}, \mathbf{y}) \sim \mu$.
Task: Sample from $(\mathbf{x} \mid \mathbf{y} = y) \sim \pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$ where posterior is unknown.
Method: Learn how to sample from posterior by solving

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\phi} \left\{ \frac{1}{m} \sum_{i=1}^{m} \left[\mathrm{D}_{\phi}(x_i, y_i) - \mathbb{E}_{\mathbf{z} \sim \sigma} \left[\mathrm{D}_{\phi}\big(\mathrm{G}_{\theta}(\mathbf{z}, y_i), y_i\big) \right] \right] \right\} \right\}$$

- Discriminator: $\mathrm{D}_{\phi} \colon X \times Y \to \mathbb{R}$.
- Generator: $\mathrm{G}_{\theta} \colon Z \times Y \to X$.
- Desired property: $\mathrm{G}_{\theta}(\mathbf{z}, y)$ approximately $\pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$-distributed when $\mathbf{z} \sim \sigma$.
- Issue: Only single sample $x_i \in X$ for each $y_i \in Y$ in training data
  $\Longrightarrow$ $\mathrm{G}_{\hat{\theta}}(\mathbf{z}, y_i) \approx \delta_{x_i}$ independent of $\mathbf{z} \sim \sigma$ (mode collapse).

Conditional Wasserstein GAN

Input: Supervised data $(x_i, y_i) \in X \times Y$ generated by $(\mathbf{x}, \mathbf{y}) \sim \mu$.
Task: Sample from $(\mathbf{x} \mid \mathbf{y} = y) \sim \pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$ where posterior is unknown.
Method: Learn how to sample from posterior by solving

$$\hat{\theta} \in \arg\min_{\theta} \left\{ \max_{\phi} \left\{ \frac{1}{m} \sum_{i=1}^{m} \left[\mathrm{D}_{\phi}(x_i, y_i) - \mathbb{E}_{\mathbf{z} \sim \sigma} \left[\mathrm{D}_{\phi}\big(\mathrm{G}_{\theta}(\mathbf{z}, y_i), y_i\big) \right] \right] \right\} \right\}$$

- Discriminator: $\mathrm{D}_{\phi} \colon X \times Y \to \mathbb{R}$.
- Generator: $\mathrm{G}_{\theta} \colon Z \times Y \to X$.
- Desired property: $\mathrm{G}_{\theta}(\mathbf{z}, y)$ approximately $\pi_{\text{post}}(\mathbf{x} \mid \mathbf{y} = y)$-distributed when $\mathbf{z} \sim \sigma$.
- Issue: Only single sample $x_i \in X$ for each $y_i \in Y$ in training data
  $\implies \mathrm{G}_{\hat{\theta}}(\mathbf{z}, y_i) \approx \delta_{x_i}$ independent of $\mathbf{z} \sim \sigma$ (mode collapse).
  Solution: Allow discriminator to distinguish between unordered pairs of model parameter or random samples generated by generator.

References I

Adler, J., Lunz, S., Verdier, O., Schönlieb, C.-B., & Öktem, O. (2018a). Task adapted reconstruction for inverse problems. *ArXiv e-print*, *cs.CV*(1809.00948).

Adler, J., Lunz, S., Verdier, O., Schönlieb, C.-B., & Öktem, O. (2018b). Task adapted reconstruction for inverse problems. In *NIPS 2018 meets medical imaging: Workshop within the 32nd annual conference on neural information processing systems (NIPS 2018).* (Also available as ArXiv e-print: https://arxiv.org/abs/1809.00948)

Adler, J., & Öktem, O. (2017). Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, *33*(12), 124007 (24pp). (Special issue: 100 Years of the Radon transform)

Adler, J., & Öktem, O. (2018a). Deep Bayesian inversion: Computational uncertainty quantification for large scale inverse problems. *ArXiv e-print*, *stat.ML*(1811.05910).

References II

Adler, J., & Öktem, O. (2018b). Learned primal-dual reconstruction. *IEEE Transactions on Medical Imaging*, *37*(6), 1322–1332.

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein generative adversarial networks. *Proceedings of Machine Learning Research*, *70*, 214–223. (Proceedings of the 34th International Conference on Machine Learning)

Banert, S., Ringh, A., Adler, J., Karlsson, J., & Öktem, O. (2018). Data-driven nonsmooth optimization. *ArXiv*, *1808.00946*.

Dalca, A. V., Balakrishnan, G., Guttag, J., & Sabuncu, M. R. (2018). Unsupervised learning for fast probabilistic diffeomorphic registration. In F. A. F., J. A. Schnabel, C. Davatzikos, L. C. Alberola, & G. Fichtinger (Eds.), *21st international conference on medical image computing and computer assisted intervention MICCAI 2018* (Vol. 11070, pp. 729–738). Springer Verlag.

References III

Farabet, C., Couprie, C., Najman, L., & LeCun, Y. (2013). Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, *35*(8), 1915–1929.

Ghosal, S., & Ray, N. (2017). Deep deformable registration: Enhancing accuracy by fully convolutional neural net. *Pattern Recognition Letters*, *94*, 81–86.

Giryes, R., Eldar, Y. C., Bronstein, A. M., & Sapiro, G. (2017). Tradeoffs between convergence speed and reconstruction accuracy in inverse problems. *ArXiv e-print*, *cs.NA*(1605.09232).

Gregor, K., & LeCun, Y. (2010). Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML'10)* (pp. 399–406).

Guo, Y., Liu, Y., Georgiou, T., & Lew, M. S. (2018). A review of semantic segmentation using deep neural networks. *International Journal of Multimedia Information Retrieval*, *7*(2), 87–93.

References IV

Hammernik, K., Klatzer, E., T. ans Kobler, Recht, M. P., Sodickson, D. K., Pock, T., & Knoll, F. (2018). Learning a variational network for reconstruction of accelerated MRI data. *Magnetic Resonance in Medicine*, *79*(6), 3055–3071.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778).

Hsieh, J.-T., Zhao, S., Eismann, S., Mirabella, L., & Ermon, S. (2019). Learning neural PDE solvers with convergence guarantees. In *Seventh international conference on learning representations (ICLR 2019).*

Karpathy, A., & Fei-Fei, L. (2017). Deep visual-semantic alignments for generating image descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(4), 664–676.

References V

Li, C. Y., Liang, X., Hu, Z., & Xing, E. P. (2018). Hybrid retrieval-generation reinforced agent for medical image report generation. *ArXiv e-print*, *cs.CV*(1805.08298).

Li, H., Schwab, J., Antholzer, S., & Haltmeier, M. (2018). NETT: Solving inverse problems with deep neural networks. *ArXiv*, *1803.00092*.

Lunz, S., Öktem, O., & Schönlieb, C.-B. (2018). Adversarial regularizers in inverse problems. *ArXiv*, *1805.11572*. (NIPS 2018)

Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., & Smolley, S. P. (2016). Least squares generative adversarial networks. *ArXiv*, *1611.04076*.

Mardani, M., Gong, E., Cheng, J. Y., Vasanawala, S. S., Zaharchuk, G., Xing, L., & Pauly, J. M. (2019). Deep generative adversarial neural networks for compressive sensing MRI. *IEEE Transactions on Medical Imaging*, *38*(1), 167–179.

Oymak, S., & Soltanolkotabi, M. (2017). Fast and reliable parameter estimation from nonlinear observations. *SIAM Journal on Optimization*, *27*(4), 2276–2300.

References VI

Putzky, P., & Welling, M. (2017). *Recurrent inference machines for solving inverse problems.* Retrieved from https://openreview.net/pdf?id=HkSOlP9lg

Rizzuti, G., Siahkoohi, A., & Herrmann, F. J. (2019). *Learned iterative solvers for the Helmholtz equation.* Submitted to 81st EAGE Conference & Exhibition 2019. (Available from https://www.slim.eos.ubc.ca/content/learned-iterative-solvers-helmholtz-equation)

Romano, Y., Elad, M., & Milanfar, P. (2017). The little engine that could: Regularization by denoising (RED). *SIAM Journal on Imaging Sciences*, *10*(4), 1804–1844.

Romano, Y., Isidoro, J., & Milanfar, P. (2017). RAISR: Rapid and accurate image super resolution. *IEEE Transactions on Computational Imaging*, *3*(1), 110–125.

Schwab, J., Antholzer, S., & Haltmeier, M. (2018). Deep null space learning for inverse problems: Convergence analysis and rates. *ArXiv*, *1806.06137*.

References VII

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). OverFeat: Integrated recognition, localization and detection using convolutional networks. *ArXiv e-print*, *cs.CV*(1312.6229).

Syu, N.-S., Chen, Y.-S., & Chuang, Y.-Y. (2018). Learning deep convolutional networks for demosaicing. *ArXiv e-print*, *cs.CV*(1802.03769).

Thoma, M. (2016). A survey of semantic segmentation. *ArXiv e-print*, *cs.CV*(1602.06541).

Wolterink, J. M., Dinkla, A. M., Savenije, M. H. F., Seevinck, P. R., van den Berg, C. A. T., & Išgum, I. (2017). Deep MR to CT synthesis using unpaired data. In S. Tsaftaris, A. Gooya, A. Frangi, & J. Prince (Eds.), *Simulation and synthesis in medical imaging. SASHIMI 2017* (Vol. 10557, pp. 14–23).

Xie, J., Xu, L., & Chen, E. (2012). Image denoising and inpainting with deep neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS 2012)* (pp. 341–349).

References VIII

Zhu, B., Liu, J. Z., Cauley, S. F., Rosen, B. R., & Rosen, M. S. (2018). Image reconstruction by domain-transform manifold learning. *Nature*, *555*, 487–492.