Deep Learning for Set Operations and Plug-and-Play Denoising

Christopher Wiedeman
wiedec@rpi.edu

***Abstract*—Applied deep learning has penetrated various fields of research, creating new, data-driven models for automated tasks and acting as denoising functions in signal reconstruction. This paper reviews two works as they relate to the current state of deep learning: *Plug-and-Play Methods Provably Converge with Properly Trained Denoisers*, which investigates the convergence of denoisers in Plug-and-Play frameworks, and *Deep Sets*, which develops neural network architectures specific to set operations. The relationship between these ideas and current deep learning topics such as transformers and network stability are discussed. Additionally, several research ideas building from the work outlined in these two manuscripts are proposed, including the investigation and use of spectral normalization for network stability and the potential use of *deep set* functions in a Plug-and-Play framework.**

I. Introduction

The significance of deep learning in computer vision and signal processing has grown exponentially in recent years. While state-of-the-art methods have become sophisticated, the fundamental deep neural network (DNN) is simply a data-fitted model consisting of sequential affine transformations followed by nonlinear functions. This cascade architecture gives multi-layer networks the capacity to robustly fit complex non-linear mappings [1]. As universal approximators, DNNs have successfully been applied to various domains for both regression and classification tasks.

DNNs have been applied to many domains, but their utility can often be categorized in one of two scenarios: the first is where an input contains sufficient information to determine the desired output, but deductively theorizing a mapping is not feasible. In these cases, such as image classification, data-driven methods are preferred, and DNNs have outperformed support vector machines [2]. The second scenario is where analytical models connecting output to input exist, but the input information is either too noisy or insufficient to accurately reconstruct the output (e.g. image reconstruction from sparse sensor data). In this case, DNNs may act similarly to compressed sensing, where the network compensates for the data insufficiency by imposing *a priori* knowledge learned from the dataset [3]. It is worth noting that for poorly conditioned inverse problems, network stability is a significant issue in deep reconstruction.

The rest of this paper will observe two works which intersect the two scenarios above. The first is *Plug-and-Play Methods Provably Converge with Properly Trained Denoisers* [4], which pertains to deep learning as applied to signal reconstruction. The second is *Deep Sets* [5], which describes and tests DNN considerations for problems involving sets. Fundamentals will be reviewed for each work, as well as their theory and findings. Finally, several new ideas pertaining to these works are proposed.

A. Background in Plug-and-Play Methods

Plug-and-play (PnP) is a general framework for extracting a desired signal with undersampled or noisy data. Consider accurately recovering signal x from data D. We can frame this as maximizing the log posterior probability $\log P(x|D)$ via Bayes' theorem:

$$x = \underset{x}{\operatorname{argmax}} \frac{P(D|x)P(x)}{P(D)} = \underset{x}{\operatorname{argmax}} \log P(D|x) + \log P(x)$$

Instead of a probability maximization, we can equivalently seek to minimize the total loss between a data likelihood term f and a prior g:

$$x = \underset{x}{\operatorname{argmin}}\ f(x) + \gamma g(x)$$

Many optimization methods exist to this end, but most involve alternating between checking data consistency (reduce $f(x)$) and denoising (reduce $g(x)$). Since reducing $g(x)$ is simply denoising, [6] realized that this step does not have to be an optimization objective, and other denoisers such as non-local means [7] can instead be used. Substituting general denoisers into this iterative framework is known as "plug-and-play" (PnP), and it has experienced broad success in applied research. Despite the empirical results, little has been done to theoretically

prove *why and under what conditions* a denoiser will converge within PnP. [4] proves convergence of PnP denoisers under certain assumptions. Specifically, with denoiser H_σ, $H_\sigma - I$ should satisfy the following Lipschitz condition for some $\varepsilon > 0$ [4]:

$$\frac{||(H_\sigma - I)(x_2) - (H_\sigma - I)(x_1)||^2}{||x_2 - x_1||^2} = \varepsilon^2 \quad (1)$$

B. Convergence Criteria for PnP

The procedures involved in proving convergence can be found in the appendix of the original paper, as they are too lengthy to include here. Essentially, the authors prove convergence assuming (1) under two PNP methods: forward-backward splitting (FBS) and Douglas-Rachford splitting (DRS), which is functionally equivalent to alternating direction of method multipliers (ADMM). FBS can be compressed into the following iterative algorithm:

$$x^{k+1} = H_\sigma(I - \alpha\nabla f)(x^k) \quad (2)$$

Intuitively, this formula is alternating between a data-fitting gradient step $I - \alpha\nabla f$ and denoising step H_σ. It is then proven that this iteration is contractive (Lipschitz constant < 1) if, assuming f is μ-strongly convex, differentiable, and L-Lipschitz:

$$\frac{1}{\mu(1 + 1/\varepsilon)} < \alpha < \frac{2}{L} - \frac{1}{L(1 + 1/\varepsilon)}$$

Which exists if $\varepsilon < 2\mu/(L - \mu)$.

For DRS, the iterative formula can be compressed as:

$$x^{k+1} = \frac{1}{2}(x^k + (2H_\sigma - I)(2\,\mathrm{Prox}_{\alpha f} - I)(x^k)) \quad (3)$$

Then, it is demonstrated that these iterations are contractive if:

$$\frac{\varepsilon}{(1 + \varepsilon - 2\varepsilon)\mu} < \alpha, \quad \varepsilon < 1$$

C. Adding Stability for DNNs in PnP

Although deep learning is not inherent to PnP, DNNs are commonly used in this framework as denoisers. [4] also proposes and tests a spectral normalization technique for training deep denoisers to fit (1). Assume denoiser $H_\sigma(x) = x - R(x)$, where R is a learned residual mapping. It is easy to see that enforcing (1) is equivalent to enforcing a Lipschitz condition on R. The Lipschitz constant of a feed-forward network can be constrained by controlling the spectral norms of each layers' weight parameters. Assuming the activation functions are non-expansive, then the Lipschitz condition of a single layer transformation corresponds to the largest singular value of the weight matrix, also known as the spectral norm. Consequently, the Lipschitz condition of the entire network is bounded by the product of spectral norms over all feed-forward layers. In other words:

$$L \leq \prod_{\ell=1}^{K} \max(\sigma(W_\ell)) \quad (4)$$

Where $\sigma(W_\ell)$ extracts the singular values of from the ℓ^{th} layer weight matrix.

Spectral normalization techniques in deep learning were first popularized with SN-GAN to stabilize discriminator training in generative adversarial learning [8]. In short, each weight matrix is routinely normalized by its largest singular value. Unfortunately, singular value decomposition (SVD) is prohibitively expensive for this use. Instead, the most significant left (u_ℓ) and right (v_ℓ) singular vectors are estimated for each W_ℓ via power iteration, which essentially estimates the leading left and right singular vectors by iteratively transforming initializations by W_ℓ or $W_\ell^\top$:

$$v_\ell^{k+1} = W_\ell^\top u_\ell^k / ||W_\ell^\top u_\ell^k||_2$$
$$u_\ell^{k+1} = W_\ell v_\ell^{k+1} / ||W_\ell v_\ell^{k+1}||_2$$

These estimations are in turn used to calculate $\sigma(W_\ell) = u_\ell^\top W_\ell v_\ell$ and normalize each transform by its spectral norm.

[8] introduces a relaxation for applying this to convolutional layers, in which the kernel is flattened, but [4] found that this method insufficient, as it both theoretically and empirically underestimates the spectral norms. As such, the authors introduce a new technique called real spectral normalization, which is analogous to power iteration for matrices but extends to convolutional kernels mapping $\mathbb{R}^{C_{in} \times h \times w} \to \mathbb{R}^{C_{out} \times h \times w}$. For convolutional kernel K_ℓ, the transpose operator K_ℓ^* is determined by permuting the first two (channel) dimensions and rotating the last two channels by $180°$ [9]. After this, the first left and right singular inputs U_ℓ and V_ℓ can be iteratively estimated in a way analogous to power iteration.

D. Pnp Experiments

[4] runs several experiments to support their theoretical findings. The most insightful is an image Poisson denoising problem. The authors test convolutional neural networks (CNNs) both with and without real spectral normalization and BM3D as denoisers in PnP-FBS and

PnP-ADMM. To assess convergence, ε was calculated for each model between iterations. It was found that $\varepsilon < 1$ for all CNNs tested, and was smaller in instances where spectral normalization was applied, guaranteeing convergence. This was not the case for B3MD, which had Lipschitz constants greater than 1. The authors also assess performance of the models via PSNR and confirm that spectrally normalized CNNs outperform BM3D. Additionally, the spectral normalized CNNs are tested in PnP algorithms for two real-world reconstructions: single-photon imaging and compressed sensing magnetic resonance imaging (MRI). In both of these experiments, the spectrally normalized CNNs yield at least competitive performance when compared with other PnP algorithms as well as other reconstruction methods, such as total variation minimization [10]. Overall, these results support the theory presented in this work, and show merit in the spectral normalization technique proposed for CNNs.

E. Neural Networks for Set Mappings

Deep Sets [5] observes machine learning tasks in which the input is a set $X = x_1, x_2, ..., x_M$, with each element x_m being an object from a domain of all possible objects (e.g. word bank). Note that the total set length M is variable. When processing sets, one of two function properties is typically desired: permutation *invariance* or permutation *equivariance*. [5] proposes deep learning approaches for both of these scenarios.

F. Permutation Invariant Deep Mappings

A function is permutation invariant if all permutations of a single input sequence produce the same output, e.g. $f(x_1, x_2, x_3) = f(x_3, x_2, x_1)$. To this end, [5] proposes a simple function form that guarantees invariance:

$$f(X) = \rho(\sum_{x \in X} \phi(x)) \tag{5}$$

The authors formally prove that all functions in this form have permutation invariance, but the intuition is self evident: elements of a sequence are first individually processed, and the summation of this process output is commutative (therefore permutation invariant), which can then be non-linearly transformed.

[5] also draws parallels between this formulation and results from other math theory, including kernel machines and spectral methods. One notable example given is de Finetti's theorem, which states that a sequence of variables are independent with regard to some latent variable given that they are exchangeable:

$$p(X|\alpha, M_0) = \int p(\theta|\alpha, M_0) \prod_{m=1}^{M} p(x_m|\theta) d\theta \tag{6}$$

In a sense, an exchangeable distribution model is an invariant set function, with the input sequence being a set. The authors specifically show that for exponential families with conjugate priors, this theorem can be simplified to the form found in Equation 5.

For deep learning set tasks, the most apparent solution is treat both ϕ and ρ as unknown non-linear functions, each of which can be approximated with feed-forward DNNs and trained end-to-end. Since ϕ processes inputs x_m individually, this structure can process sets of variable length, similar to a recurrent neural network (RNN).

G. Permutation Equivariant Deep Mappings

The second scenario considered is permutation equivariance, meaning that a certain permutation of an input sequence simply produces the same output sequence but with that same permutation (e.g. if $f(x_1, x_2, x_3) = (y_1, y_2, y_3)$ then $f(x_3, x_2, x_1) = (y_3, y_2, y_1)$). This property can only apply to $f : \mathbb{R}^M \rightarrow \mathbb{R}^M$. To this end, the authors propose the following equivariant neural network layer architecture:

$$f(x) = \sigma(\lambda \mathbf{I} x + \gamma \, \text{maxpool}(x)\mathbf{1}) \tag{7}$$

Where $\mathbf{1}$ is a vector of 1's, γ and λ are scalar parameters, and σ is an activation function. Using this structure not only makes each layer-wise transformation equivariant, but also allows for variable length inputs, as the number of parameters is independent of input size.

H. Experimental Results

The authors of [5] apply their methods to numerous real-world experiments, several of which are simple regressions or classifications with set inputs. First, various methods are used to generate multivariate Gaussian distributions, with the task of calculating entropy and mutual information within distributions only from a set of samples. A 3-layer permutation invariant (Equation 5) DeepSet model with ReLU activations was trained for this, which consistently outperformed a support distribution machine model [11]. Next, a DeepSet model is trained to sum a set of digits (both in text and image form), and achieves better results than both long short-term memory (LSTM) and gated recurrent units (GRU) models [12] [13]. Additionally, the authors demonstrate competitive performance on point cloud classification

tasks (classifying an object from a set of points forming a 3D mesh) and estimation of galaxy red-shift from photometrics.

An unsupervised, more interesting tested task is set expansion, in which a model is given a set of inputs an must predict another object that fits well within this set. If one considers a set as a point in an exchangeable model with de Finetti's theorem (Equation 6), then one can also view the expansion task as selecting the object x with maximum point-wise mutual information with input X:

$$s(x|X) = \log p(X \cup x|\alpha) - \log p(X|\alpha)p(x|\alpha) \quad (8)$$

This framing is used to create a relative loss function for network training:

$$l(x, x'|X) = \max(0, s(x'|X) + \Delta(x, x') - s(x|X))$$

Where x is an expansion that feasibly exists in X, and x' does not feasibly exist. The authors also note that data-driven DNNs can easily incorporate meta-data (e.g. an image associated with tags), unlike other existing models. To this end, DeepSets is applied successfully when in both text concept set retrieval (predicting a word that shares conceptual similarity with words in an input set) and image tagging (adding additional appropriate tags to an image).

Finally, the authors apply their equivariant DNN model to anomaly detection (detecting an object that does not belong in a set) in face images. The tested network consisted of a convolutional network as a feature extractor followed by a sequence of either fully-connected or permutation equivariant layers. Use of the equivariant layers achieved better performance, confirming its utility in appropriate tasks.

Overall, Deepsets proposes and tests robust deep learning methods for set operations when either invariance or equivariance is desired. The theory provided gives sound explanation for the architectural choices, and experimental results validate their advantages.

II. Discussion

Both *Deep Sets* and *Plug-and-Play Methods* explore relevant challenges in deep learning, providing both theoretical and empirical findings for their ideas. The following section explores potential research directions based on the ideas presented and how they relate to broader topics in the field of deep learning. Namely, observing network spectral norms as they relate to stability and adversarial robustness and as a broader regularization method, Deep Sets' relation to other current architectures, and an application that incorporates both Deep Sets and PnP methods are discussed.

A. Spectral Norms for Robustness and Regularization

Plug-and-Play Methods guarantees PnP convergence by limiting the Lipschitz constant of the denoiser network and provides a powerful method for spectral normalization in convolutional layers. Controlling the Lipschitz constant of a DNN is a hot topic, especially in adversarial robustness, where it is thought that instability against adversarial attacks are related to large Lipschitz constants [14]. *The Troublesome Kernel* explores instability in deep medical image reconstruction, and even claims that such Lipschitz instability is inherent in the problem due to a lack of kernel awareness [15].

Spectral normalization proposes a potential method for controlling the Lipschitz constant, but far more investigation is needed to prove its value. Equation 4 only gives an upper limit on the network's Lipschitz constant; it does not guarantee that any input that could generate such a divergence exists, let alone whether or not that input feasibly exists within the input distribution. It is also unclear when spectral norms should be controlled for stability. For example, *Figure 1* of [4] shows that even networks without spectral normalization achieved $\varepsilon < 1$, suggesting that the network learned a stable mapping on its own.

Whether or not higher spectral norms truly lead to more instability should be explored. This would be easy to evaluate in the context of adversarial attacks, as one could train various classification networks and observe their adversarial robustness in relation to layer spectral norms. Furthermore, if large spectral norms cause adversarial instability, then one would expect adversarial examples to have *significant components along the singular vectors of at least one of the layer transformations during forward propagation*. In other words, given training example x and adversarial example $x' = x + \Delta x$, for some layer ℓ, the different in hidden values $h'_\ell - h_\ell$ should be very closely aligned with one of W_ℓ's largest righthand singular vectors. This can be easily tested.

If results show that high spectral norms indeed cause adversarial instability, then regularization of spectral norms should be investigated. Current spectral normalization methods simply normalize the each layer's spectral norm to 1, but this is a very rigid application, as some mappings may necessarily require larger singular values.

This approach also only scales each singular value evenly, rather than reducing the largest singular value directly. Using the power iteration method mentioned earlier with left and right singular vectors u_ℓ and v_ℓ, one can easily see that the gradient of parameters W_ℓ with respect to the layer's spectral norm is:

$$\nabla_{\sigma(W_\ell)} W_\ell = u_\ell v_\ell^\top \tag{9}$$

Given this, the following regularization is proposed during training for each layer:

$$\begin{aligned} W_\ell^{k+1} &= W_\ell^k - \eta(\nabla_{\mathcal{L}} W_\ell + \lambda \nabla_{\sigma(W_\ell)} W_\ell) \\ \nabla_{\sigma(W_\ell)} W_\ell &= u_\ell v_\ell^\top \;\; if \;\; \sigma(W_\ell) > 1 \\ &= 0 \;\; o.w. \end{aligned}$$

Where $\mathcal{L}$ is some loss function. Employing this regularization penalizes transformations only if they are potentially expansive, but still does not disallow expansive transforms entirely, facilitating convergence towards a mapping that is both accurate and stable.

B. Deep Sets in Relation to other Architectures

It is worth noting that while other works have perhaps not explored the theoretical realm of deep set processing as much as [5], many have arrived at related architectures to the permutation invariant approach. The transformer is the most prominent example of this, which has shown great success in natural language processing [16]. Similar to the permutation invariant approach in [5], the transformer first processes inputs individually into embeddings. Rather than explicitly summing outputs, however, the transformer uses self-attention to facilitate interaction between inputs, which is invariant of sequence permutation. Language processing is often *not* a set task, as the order of words is important to the embedded meaning, so the transformer typically adds a positional encoding to each embedded input to preserve sequence information. For complex set operations, the obvious architecture would simply be a transformer without this positional encoder. Given the transformer's success, especially with language tasks, it would reason that such an architecture would perform well in set tasks like expansion. Perhaps architectures that process inputs invariantly offer generally more robust frameworks, as positional encodings can be added or removed depending on the nature of the problem.

C. Deep Sets in PnP for Reconstruction with Noisy Sampling

One possible application of both PnP methods and Deep Sets would be a reconstruction task for noisy data that can represent a set. For instance, imagine a scenario where a sensor array can record more than sufficient data, but reconstruction is extremely noisy, and the noise model is either complicated not clearly defined. An example of this would be Compton Tomography imaging, which is similar to Computed Tomography (CT), but instead reconstructs images with incoherently scattered energy rather than transmission [17]. It is theoretically possible reconstruct an image detecting only a single scatter energy with collimated detectors, but the data received is extremely noisy due to the effects of an unknown attenuation map. One solution to this is to jointly reconstruct the attenuation image and scatter image [18] [19]. Using scatter information from multiple energies can help constrain this reconstruction, but simultaneously considering all scatter energies in a single data fidelity term is difficult to optimize. In this instance, one can separate the recorded data by scatter energy and reconstruct multiple, noisy images and then feed these reconstructions into a DNN denoiser as a set of inputs. The denoiser would consider the input as a set of images, with the task of outputting a single, denoised image. This approach could be particularly useful if the noise model is not well defined. Such an iterative reconstruction should include consistency between the different reconstructions outputs of the data fidelity step as part of the convergence criteria.

III. Conclusion

This work has reviewed the fundamentals presented in *PnP Algorithms* and *Deep Sets*, two works broadly related to deep learning. As both present strong theoretical exploration as well as empirical results, new research can either explore the theory or apply the ideas presented to specific domains. In particular, a more theoretical exploration into spectral normalization as it would relate to network instability, the relationship between the *Deep Sets* permutation invariant architecture and popular transformer architecture, and a potential application of the invariant architecture and PnP methods to scenarios where data can be abundantly sampled, but with difficult underlying noise models are all discussed.

References

[1] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[2] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Neural Information Processing Systems*, vol. 25, 01 2012.
[3] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.
[4] E. K. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, and W. Yin, "Plug-and-play methods provably converge with properly trained denoisers," 2019.
[5] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.
[6] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *2013 IEEE Global Conference on Signal and Information Processing*, 2013, pp. 945–948.
[7] A. Buades, B. Coll, and J. . Morel, "A non-local algorithm for image denoising," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, 2005, pp. 60–65 vol. 2.
[8] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," 2018.
[9] J. Liu, X. Chen, Z. Wang, and W. Yin, "Alista: Analytic weights are as good as learned weights in lista," in *ICLR*, 2019.
[10] M. Lustig, J. Santos, J.-H. Lee, D. Donoho, and J. Pauly, "Application of "compressed sensing" for rapid mr imaging," 01 2005.
[11] B. Póczos, L. Xiong, D. J. Sutherland, and J. Schneider, "Support distribution machines," *ArXiv*, vol. abs/1202.0302, 2012.
[12] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997.
[13] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 2014.
[14] C. Finlay, A. M. Oberman, and B. Abbasi, "Improved robustness to adversarial examples using lipschitz regularization of the loss," 2019. [Online]. Available: https://openreview.net/forum?id=HkxAisC9FQ
[15] N. Gottschling, V. Antun, B. Adcock, and A. Hansen, "The troublesome kernel: why deep learning for inverse problems is typically unstable," *ArXiv*, vol. abs/2001.01258, 2020.
[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.
[17] T. Truong and M. Nguyen, "Recent developments on compton scatter tomography: Theory and numerical simulations," in *Numerical Simulation - From Theory to Industry*, 2012, pp. 101–128.
[18] M. R. Walker II and J. A. O'Sullivan, "Iterative algorithms for joint scatter and attenuation estimation from broken ray transform data." [Online]. Available: http://arxiv.org/abs/2006.14719
[19] H. Rezaee, B. Tracey, and E. Miller, "On the fusion of compton scatter and attenuation data for limited-view x-ray tomographic applications," *ArXiv*, vol. abs/1707.01530, 2017.