ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

LEAP nets for power grid perturbations

B. Donnot[†‡*], B. Donon[†‡], I. Guyon[•‡], Z. Liu[‡],
A. Marot[†], P. Panciatici[†], M. Schoenauer[‡]
• ChaLearn, USA. ‡ UPSud/Inria, U. Paris-Saclay, France. † RTE France

Abstract. We propose a novel neural network embedding approach to model power transmission grids, in which high voltage lines are disconnected and re-connected with one-another from time to time, either accidentally or willfully. We call our architecture LEAP net, for Latent Encoding of Atypical Perturbation. Our method implements a form of transfer learning, permitting to train on a few source domains, then generalize to new target domains, without learning on any example of that domain. We evaluate the viability of this technique to rapidly assess curative actions that human operators take in emergency situations, using real historical data, from the French high voltage power grid.

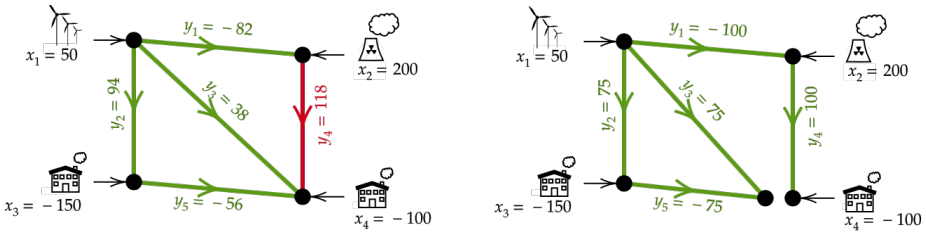


Fig. 1: Electricity is transported from production nodes (top) to consumption nodes (bottom), through lines (green and red edges) connected at substations (black circles), forming a transmission *grid* of a given topology $\boldsymbol{\tau}$. Injections $\boldsymbol{x} = (x_1, x_2, x_3, x_4)$ (production or consumption) add up to zero. Grid operators (a.k.a. *dispatchers*) should maintain current flows $\boldsymbol{y} = S(\boldsymbol{x}, \boldsymbol{\tau})$ below thermal limits. Left: Line y_4 goes over its thermal limit 100. Right: A change in topology (splitting of node 6) brings y_4 back to its thermal limit.

1 Background and motivations

We address the problem of accelerating the computation of current flows in power transmission grids, using artificial neural networks, to emulate slower physical simulators, following other pioneering work [6, 5, 3, 2]. Key to our approach is the possibility of simulating the effect of planned coordinated actions on the grid topology (as opposed to accidental suffered changes). Our neural network models may then be used as part of an overall computer-assisted decision process in which human operators (dispatchers) ensure that the power grid is operated in security at all times, namely that the currents flowing in all lines are below certain thresholds (line thermal limits). Figure 1 illustrates the problem setting on a toy example. If one line goes over its thermal limits, it may be damaged, melt and/or cause fire or break, thus circuit breakers usually put it out of service before this happens. Hence, the grid must be reconfigured quickly to re-balance

*Benjamin Donnot corresponding authors: benjamin.donnot@inria.com

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

current flows and avoid that more lines go over their thermal limit, which might result in a cascading effect (black-out). The space of possible grid topologies grows exponentially with the number of substations. For example, the French high-voltage transmission grid includes $N \approx 6200$ substations, with more than a dozen possible configurations per substation and thus $\gtrapprox 10^N$ possible grid topologies. Even if only a small number of those are achievable, the search space is still humongous. In practice, Transmission System Operators (TSOs) limit dispatchers to a very limited set of candidate operations. However, operating the grid is becoming increasingly complex because of the advent of less predictable renewable energies, the globalization of energy markets, growth in consumption and concurrent limitations on new line construction. Therefore, it is becoming urgent to optimize more tightly the grid operation, considering a broader range of topological changes operated more frequently, without compromising security.

2 Proposed methodology

Our objective is to approximate a function $\boldsymbol{y} = S(\boldsymbol{x}, \boldsymbol{\tau})$ that maps input data $\boldsymbol{x}$ (*e.g.* power production and consumption) to output data $\boldsymbol{y}$ (*e.g.* power flows), parameterized by a discrete "grid topology vector" $\boldsymbol{\tau}$, taking values in an action space (all possible power-grid topologies *e.g.* line interconnections). For any fixed topology $\boldsymbol{\tau}$, training data pairs $\{\boldsymbol{x}, \boldsymbol{y}\}$ are drawn *i.i.d.* according to an unknown probability distribution. In our application setting, $\boldsymbol{x}$ is drawn randomly, but $S(\boldsymbol{x}, \boldsymbol{\tau})$ is a deterministic function implementing Kirchhoff's circuit laws, calculated by a physical simulator that we wish to approximate.

We call **simple generalization** the capability of a neural net $\hat{\boldsymbol{y}} = NN(\boldsymbol{x}, \boldsymbol{\tau})$ to approximate $\boldsymbol{y} = S(\boldsymbol{x}, \boldsymbol{\tau})$ for test inputs $\boldsymbol{x}$ not pertaining to the training set, when $\boldsymbol{\tau}$ values are drawn *i.i.d.* from a distribution that remains the same in training and test data (this includes the case of a fixed $\boldsymbol{\tau}$). Conversely, if values of $\boldsymbol{\tau}$ are drawn according to a **source domain** distribution in training data and from a different **target domain** distribution in test data, then we will talk about **super-generalization**. This setting is a particular case of transfer learning [7].

One particularity of our application domain in terms of transfer learning is that we have one *primary* "reference" source domain (corresponding in the power grid to a reference grid topology $\boldsymbol{\tau}^{\emptyset} = (0, 0, 0, \dots)$, around which *small* variations are made. This is a generic scenario in the industry for systems that operate around nominal conditions, thus we anticipate that our method could be extended to other similar situations. In our application setting, we can easily get a lot of training data in the reference topology (corresponding to the typical way in which the grid is operated). We have comparably very little data available for training from other *secondary* source domains, corresponding to unary changes in grid topology $\boldsymbol{\tau}^{i} = (0, 0, 1, \dots)$ (a single 1 at position i). Finally, we have extremely scarce data or no data at all available for training from domains corresponding to double changes $\boldsymbol{\tau}^{ij}$, or higher order changes (considered target domains). This motivates our architectural design.

Our proposed Latent Encoding of Atypical Perturbations network, or LEAP

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

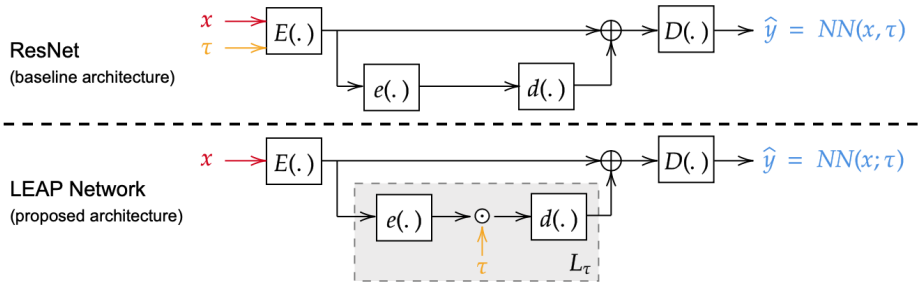


Fig. 2: **Baseline and LEAP architectures:** Top: ResNet [4] architecture, with $\boldsymbol{\tau}$ as input. Bottom: Proposed LEAP net: $\boldsymbol{\tau}$ intervenes in the latent embedding space. The effect is to make a "leap" in latent space.

net (Figure 2), is composed of three parts: An Encoder $\boldsymbol{E}$, learning an embedding of the input data x; a Decoder $\boldsymbol{D}$, learning how to perform the required task within this latent representation; and a Latent module $\boldsymbol{L}_\tau$, placed between the $\boldsymbol{E}$ and $\boldsymbol{D}$ where $\boldsymbol{\tau}$ intervenes. The overall arhitecture is given by:

$$\boldsymbol{L}_\tau \quad : \quad h \quad \rightarrow \quad \boldsymbol{d}(\boldsymbol{e}(h) \odot \boldsymbol{\tau}) \tag{1}$$

$$\hat{y} \quad = \quad \boldsymbol{D} \circ (\boldsymbol{I} + \boldsymbol{L}_\tau) \circ \boldsymbol{E}(x) \tag{2}$$

where $\boldsymbol{E}$ and $\boldsymbol{e}$ (encoders) and $\boldsymbol{D}$ and $\boldsymbol{d}$ (decoders) are all differentiable functions (typically implemented as artificial neural networks). The $\odot$ operation denotes the component-wise multiplication and $\circ$ the function composition. If the system is in the reference topology $\boldsymbol{\tau}^\emptyset$, predictions are made according to $\hat{\boldsymbol{y}} = \boldsymbol{D} \circ \boldsymbol{E}(\boldsymbol{x})$. A typical way in which we train LEAP nets is to use a lot of training data in the reference topology $\boldsymbol{\tau}^\emptyset$ (primary source domain), very few examples for each of the unary changes $\boldsymbol{\tau}^i$ (secondary source domains), and we expect the network to generalize to target domains corresponding to double $\boldsymbol{\tau}^{ij}$ or higher level changes.

While our architecture draws inspiration from both Dropout [8] and Residual Neural Networks [4], in its mathematical formulation, the underlying concept is quite different. Here we first embed x in a latent space by applying $\boldsymbol{E}(x)$. Then, based on $\boldsymbol{\tau}$ and the location of $\boldsymbol{E}(x)$ within the latent space, we compute the corresponding leap $\boldsymbol{L}_\tau \circ \boldsymbol{E}(x)$. Then we decode the signal by applying $\boldsymbol{D}$. Those latent leaps contain information about how much the system actually deviates from the reference state, and in which direction. Hence, our architecture only needs to learn to modulate the system response around its nominal value.

3 Predicting flows in power grids

We present results for our target application on simulated and real data. Synthetic data allows us to perform controlled systematic experiments and compare neural network approaches with a standard baseline (DC approximation) in power systems. Real data allows us to check whether our method scales computationally while providing prediction accuracies that are acceptable for our application domain.

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

3.1 Case 118 synthetic data benchmark

We conducted controlled experiments on a standard medium-size benchmark from "Matpower" [9], a library commonly used to test power system algorithms [1]: case118, a simplified version of the Californian power grid (dim $\boldsymbol{x} = 153$ injections and dim $\boldsymbol{y} = 186$ power lines). Topology changes consist in reconfiguring line connections in one or more substations (see Figure 1). Such changes are more complex than simple line disconnections considered in [3]. There are 11 558 possible unary actions (corresponding to single node splitting or merging, compared to the reference topology). To build the Source domain training and test sets, we sampled randomly 100 $\boldsymbol{\tau}^{(i)} \in \mathcal{T}^{Source}$. In the reference topology ($\boldsymbol{\tau}^{\emptyset}$), we sampled 50000 input vectors $\boldsymbol{x}$. But for each $\boldsymbol{\tau}^{(i)}$, we sampled only 1000 input vectors $\boldsymbol{x}$. We used Hades2[1] to compute the flows $\boldsymbol{y}$ in all cases. This resulted in a training set of 150 000 rows (each row being one triplet $(\boldsymbol{x}, \boldsymbol{\tau}^{(i)}, \boldsymbol{y})$). We created an independent test set of the same size in a similar manner.

We proceeded differently for the Target dataset. We sampled 1500 (Target domains: $\boldsymbol{\tau}^{(ij)} \in \mathcal{T}^{Target}$) among the 4950 possible double actions $\boldsymbol{\tau}^{(ij)} = \boldsymbol{\tau}^{(i)} \vee \boldsymbol{\tau}^{(j)}$, $\boldsymbol{\tau}^{(i)}$ and $\boldsymbol{\tau}^{(j)} \in \mathcal{T}^{train}$. Then, for each of these 1500 $\boldsymbol{\tau}^{(ij)}$, we sampled 100 inputs $\boldsymbol{x}$ (with the same distribution as the one used for the training and regular test set). We used the same physical simulator to compute the $\boldsymbol{y}$ from the $\boldsymbol{x}$ and the $\boldsymbol{\tau}$. The super-generalization set counts then 150 000 rows, corresponding to 150 000 different triplets $(\boldsymbol{x}, \boldsymbol{\tau}^{(ij)}, \boldsymbol{y})$.

We compare the proposed LEAP net with two benchmarks: the DC approximation, a standard baseline in power systems, which is a linearization of the AC (Alternative Current) non-linear powerflow equations, and the baseline neural network architecture (Figure 2) in which $\boldsymbol{\tau}$ is simply an input. The mean-square error was optimized using the Tensorflow Adam optimizer. To make the comparison least favorable to LEAP net, all hyper-parameters (learning rates, number of units) were optimized by cross-validation for the baseline network.

Figure 3 indicates that the LEAP net (blue curves) performs better than the DC approximation (black line) both for regular and super generalization. Figure 3b shows that the baseline neural network architecture (green curve) is not viable: not only does it perform worse than the DC approximation, but its variance is quite high. While it is improving in regular generalization with the number of training epochs, its super-generalization performances get worse.

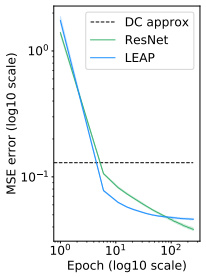
3.2 Real French ultra-high voltage power grid data

We now present results on a part of the French ultra-high voltage power grid: the "Toulouse" area with 246 consumption nodes, 122 production nodes, 387 lines and 192 substations often split in a variable number of nodes. The inputs $\boldsymbol{x}$ representing injections (production and consumption) are of dim $\boldsymbol{x} = 368$) and the outputs $\boldsymbol{y}$ (flows) of dim $\boldsymbol{y} = 387$. In this study, $\boldsymbol{x}$ and $\boldsymbol{y}$ come from real historical data from the company RTE[2]. One important difference when using

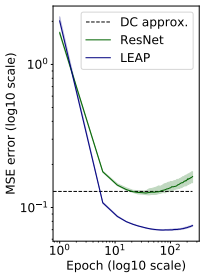
[1] Freeware available at `http://www.rte.itesla-pst.org/`.

[2] Even in real records, flows are estimated, not measured.

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

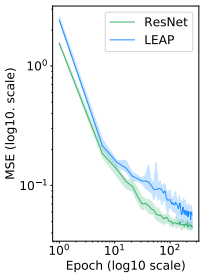


(a) Regular gene.

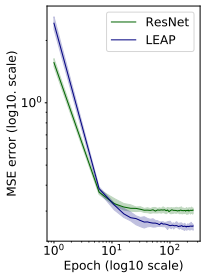


(b) Super- gene.

Fig. 3: **Synthetic data (case 118).** Neural nets trained with 15000 injections, for $\boldsymbol{\tau}^{\emptyset}$ and unary changes $\boldsymbol{\tau}^{(i)}$. (a) **Regular generalization.** Test injections for **unary changes** $\boldsymbol{\tau}^{(i)}$. (b) **Super-generalization.** Test injections for **double changes** $\boldsymbol{\tau}^{(ij)}$. Error bars are [20%, 80%] intervals, computed over 30 repeat experiments.



(a) Regular gene.



(b) Super-gene.

Fig. 4: **Real data from the ultra high voltage power grid.** The neural net in both cases is **trained from data until May 2017**. (a) **Regular generalization.** Test set made of randomly sampled data in same time period as training data. (b) **Super-generalization.** Test set made of the months of June and July 2017.

played-back data, compared to simulation, is that we cannot intervene (this is strictly observational data). To place ourselves in a realistic transfer learning setting, we used data from 2012 to May 2017 for $\mathcal{T}^{Source}$ and data from June and July 2017 for $\mathcal{T}^{Target}$. This favored changes in $\boldsymbol{\tau}$ distribution. Another key difference in real data is "actions space". In real data *actual* grid topologies (specifying line interconnections) are not precisely recorded. Only information on *line outages* is available to us as surrogate information on topology. This makes the neural net task harder: it must learn the effects of latent topological changes. This unfortunate loss of information on exact grid topology interventions makes it impossible for us to compare our method to the DC approximation: computing this approximation requires a full description of the topology. The results of Fig. 4 yield the same conclusions as in the previous section: the LEAP model generalizes not only to data drawn from a similar distribution it was trained on (Fig. 4a) but also to unseen grid states (Fig. 4b), better than the reference architecture, which is a critical property for our application.

4 Discussion and conclusion

The LEAP net architecture has been evaluated on a number of real and artificial test cases. Training was performed on data triplets $(\boldsymbol{x}, \boldsymbol{\tau}, \boldsymbol{y})$, for which $\boldsymbol{\tau} \in \mathcal{T}^{Source}$ belong to source domains. The LEAP net generalizes not only by approximating well $\boldsymbol{y}$ for new values of $\boldsymbol{x}$ when $\boldsymbol{\tau} \in \mathcal{T}^{Source}$, but also when $\boldsymbol{\tau} \in \mathcal{T}^{Target}$ (super-generalization). In our experiments, we achieved a speed-up of ≈ 300 times using the LEAP net, compared to running the physical simulator, on the synthetic dataset (power grid of 118 nodes). With data stored in computer memory, our experiments on the Toulouse area attain a speed of $\simeq 2000$ times compared to running the physical simulator. These computational

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

evaluations were carried out using a single high-end Graphical Processing Unit (GPU) Nvidia Titan X. Further work includes scaling up our method computationally to the entire French extra high voltage power grid. We also need to improve prediction accuracy before our system could be deployed to production. However, the fact that the regular generalization performance is already within an acceptable accuracy range shows great promises. We anticipate several developments. From the theoretical point of view, we could seek mathematical guarantees of super-generalization in the form of performance bounds. It can easily be proved that a LEAP net architecture with linear submodules $\boldsymbol{d}$ and $\boldsymbol{D}$ exhibits super-generalization with respect to linear superposition of perturbations. However, we have demonstrated experimentally that super-generalization extends to combinations of non-linear perturbations. We are hopeful that more powerful theoretical results could be derived. From the practical point of view, the LEAP net architecture could be used in other application domains, lending themselves to transfer learning.

References

[1] O. Alsac and B. Stott. Optimal load flow with steady-state security. *IEEE transactions on power apparatus and systems*, PAS-93(3):745–751, 1974.

[2] B. Donnot, I. Guyon, M. Schoenauer, A. Marot, and P. Panciatici. Anticipating contingengies in power grids using fast neural net screening. In *IEEE WCCI 2018*, Rio de Janeiro, Brazil, July 2018.

[3] B. Donnot, I. Guyon, M. Schoenauer, A. Marot, and P. Panciatici. Fast power system security analysis with guided dropout. In *ESANN*, Apr. 2018.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, pages 630–645. Springer, 2016.

[5] T. Hossen, S. J. Plathottam, R. K. Angamuthu, P. Ranganathan, and H. Salehfar. Short-term load forecasting using deep neural networks (dnn). In *2017 North American Power Symposium (NAPS)*, pages 1–6, Sept 2017.

[6] T. Nguyen. Neural network load-flow. *IEE Proceedings - Generation, Transmission and Distribution*, 142:51–58(7), January 1995.

[7] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knoweledge and Data Engineering*, 22(10):1345–1359, October 2010.

[8] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.

[9] R. D. Zimmerman and et al. Matpower. *IEEE Trans. on Power Systems*, pages 12–19, 2011.