Adversarial Dictionary Learning

Jordan Frecon[1], Lucas Anquetil[1], Gilles Gasso[1], and Stéphane Canu[1]

[1]Normandie Univ, INSA Rouen, UNIROUEN, UNIHAVRE, LITIS.
Saint-Etienne-du-Rouvray, France

June 15, 2021

Abstract

This work frames the learning of multiple adversarial perturbations as a sparse dictionary learning problem bridging the gap between specific and universal attacks. On the one hand, this framework allows to build an adversary attack to new examples by only learning the coding vectors, provided that the dictionary is known. On the other hand, the *a posteriori* study of the atoms unveils the most common patterns to attack the classifier. Numerical experiments conducted on CIFAR-10 illustrate that our approach, termed as Sparse Coding of ADversarial Attacks (SCADA), achieves higher fooling rates of the deep model than state-of-the-art attacks for smaller adversarial perturbations.
Keywords: adversarial; sparse coding.

1 Introduction

With recent technological advances, the use of deep neural networks (DNN) have widespread to numerous applications ranging from biomedical imaging [MLY17] to the design of autonomous vehicles [BACM19]. The reasons of their prosperity strongly rely on the increasingly large datasets becoming available, their high expressiveness and their empirical successes in various tasks (e.g. computer vision [AM18], natural language processing [YHPC18] or speech recognition [DHK13]).

However, their high representation power is also a weakness that some adversary might exploit to craft adversarial attacks which could potentially lead the DNN model to take unwanted actions [SZS+14, FBI+19]. More precisely, adversarial attacks are almost imperceptible transformations aiming to modify an example well classified by a DNN into a new example, called adversarial, which is itself wrongly classified. To date, various more or less heuristic attacks have been developed. The majority of them are perturbations which, added to the original image, will change few pixels. In this regard, the most popular are the fast gradient sign method (FGSM) [GSS15, KGB17], DeepFool [MFF16], the projected gradient method (PGD) [MMS+19] or the approach of Carlini and Wagner (CW) [CW17] which relies on the minimization of the perturbation's ℓ_p-norm. More recently, a functional attack, called ReColorAV, has been devised in [LF19] and intends to learn a perturbation function while applied to the input produces an adversarial example. A peculiarity of all these attacks is that they are *specific*, meaning that they are intended to modify a single example. A contrario, in [MDFFF17] the authors devised a *universal* perturbation, coined UAP, that can be applied to a whole set of images. However, although it is universal, it is difficult to interpret precisely why it works in a case to case basis. More generally, state-of-the-art attacks still suffer from a lack of interpretability.

This contribution aims to bridge the gap between specific and universal attacks through adversarial dictionary learning by allowing each individual attack to be written as a linear combination of a few shared dictionary elements. The proposed framework is introduced in Section 2 while a corresponding algorithmic solution is devised in Section 3. Numerical experiments are conducted in Section 4 where the dataset CIFAR-10 has been used as a challenging reference point of evaluation for the proposed algorithm.

Notations. Let $\mathcal{X}$ be a Hilbert space and $\Gamma_0(\mathcal{X})$ be the set of proper lower semicontinuous convex functions from $\mathcal{X}$ to $]-\infty,+\infty]$. We let $\imath_{\mathcal{C}}$ denotes the indicator function of the set $\mathcal{C} \subseteq \mathcal{X}$, i.e., $\imath_{\mathcal{C}}(x) = 0$ if $x \in \mathcal{C}$ and $+\infty$ otherwise. The proximity operator of $h \in \Gamma_0(\mathcal{X})$ reads $\text{prox}_h \colon u \in \mathcal{X} \mapsto \text{argmin}_{v\in\mathcal{X}} \frac{1}{2}\|u-v\|^2 + h(v)$.

2 Learning dictionary of attacks

In this section, we introduce the proposed framework for learning a dictionary of attacks.

Without loss of generality, we focus on adversary examples aiming to fool a classifier in the form of a generic neural network f with $c \in \mathbb{N}^+$ output nodes and taking an image x as input. Hence, for every image $x \in [0,1]^P$ made of P pixels, $f(x) \in \mathbb{R}^c$. The predicted label for x by the classifier f is then defined as the maximizer $C_f(x) = \operatorname{argmax}_{k=1,\ldots,c} f_k(x)$. The goal of adversarial learning is to find an example x' close to x so that the predicted labels are different, i.e., $C_f(x') \neq C_f(x)$. Here, we restrict to adversary examples written as additive perturbations of valid examples. In this setting, the goal is to find a small perturbation ϵ such that $x' = x + \epsilon$ is an adversary example.

The originality of the proposed method is to find a perturbation ϵ expressed as a linear combination of a few attacks. More formally, let μ be a distribution of images on $[0,1]^P$. We aim to learn a dictionary of attacks $D \in \mathbb{R}^{P\times M}$, made of $M \ll P$ atoms, so that for every $x_i \sim \mu$, $x_i + \epsilon_i$, with $\epsilon_i = Dv_i$, is an adversary example for some sparse vector $v_i \in \mathbb{R}^M$. To do so, we propose to address the following optimization problem reminiscent of classical dictionary learning problems (see, e.g., [MPS+09, Rak13]).

Problem 2.1 (Adversarial dictionary learning)**.** Let a classifier $f\colon \mathbb{R}^P \to \mathbb{R}^c$ and a dataset $\{x_i, t_i\}_{i=1}^N$ made of $N \in \mathbb{N}^+$ samples where each $x_i \in \mathbb{R}^P$ is a valid instance and $t_i \in \mathbb{R}$ is an adversary target chosen amongst the c classes so that $t_i \neq C_f(x_i)$. Solve

$$\underset{\substack{D\in\mathcal{C}\\ V=[v_1\cdots v_N]\in\mathbb{R}^{M\times N}}}{\text{minimize}} \sum_{i=1}^N \ell_i(D, v_i), \qquad (1)$$

where $\mathcal{C} = \{D \in \mathbb{R}^{P\times M} \mid (\forall m \in \{1,\ldots,M\}),\ \|d_m\|_2 \leq 1\}$ encodes some bounding constraints on D and where the cost associated to each adversary attack reads

$$\ell_i(D, v_i) = \lambda_1\|v_i\|_1 + \lambda_2\|Dv_i\|_2^2 + H(f(x_i + Dv_i), t_i).$$

H represents the cross-entropy loss while $\lambda_1, \lambda_2 > 0$ are regularization parameters.

The first term in the cost $\ell_i(D, v_i)$ enforces v_i to be sparse through the use of the ℓ_1-norm. As a result, only a few atoms of D are chosen to build the perturbation Dv_i. The second term favors an adversary example $x_i + Dv_i$ close to x_i with respect to the ℓ_2-norm. The third term measures the cross-entropy between the output of the classifier fed with the adversary example and the adversary target. It plays a central role since it quantifies the closeness between the predicted target and the adversary target.

3 Algorithmic solution

Herein we propose a procedure for solving Problem 2.1.

Minimizing the objective in (1) is a challenge due to the nonconvexity inherent to the dictionary learning formulation and the neural network f. We stress out that we are only interested in finding a good stationary point in a limited time. Although classical dictionary learning problems are nonconvex, they are usually solved by alternating the optimization over D and V since each alternating problem is convex. However, here this no longer the case because of the added term promoting adversarial examples. Hence, we embrace a direct optimization scheme over (D, V) in the spirit of the nonconvex proximal splitting framework of [Sra12] which has also been applied in the context of classical dictionary learning in [Rak13]. To that purpose, we begin by recasting the objective in Problem 2.1 as follows.

$$\underset{\substack{D\in\mathbb{R}^{P\times M}\\ V\in\mathbb{R}^{M\times N}}}{\text{minimize}}\ \mathcal{L}(D,V) \triangleq \Big\{F(D,V) + \Omega(D,V)\Big\}, \qquad (2)$$

with

$$\begin{cases} F(D,V) &= \sum_{i=1}^N \lambda_2\|Dv_i\|_2^2 + H\big(f(x_i + Dv_i), t_i\big),\\ \Omega(D,V) &= \imath_{\mathcal{C}}(D) + \sum_{i=1}^N \lambda_1\|v_i\|_1, \end{cases}$$

where F is smooth, provided that f is smooth as well, and $\Omega \in \Gamma_0(\mathbb{R}^{P\times M} \times \mathbb{R}^{M\times N})$ is nonsmooth. We also make the assumption that ∇F is Lipschitz continuous. The existence of the Lipschitz constant plays a central role for ensuring convergence guarantees of the algorithm. Note that studying the Lipschitz regularity of neural networks is difficult [SV18]. Here, we promote the linesearch based proximal-gradient method of [BLP+17]. For the ease of reading, in what follows we only focus on the proximal-gradient step. Given some sequence of step-sizes $\{\gamma_k\}_{k\in\mathbb{N}}$, each step amounts in finding

$$(D^{(k+1/2)}, V^{(k+1/2)}) = \underset{D\in\mathbb{R}^{P\times M}, V\in\mathbb{R}^{M\times N}}{\operatorname{argmin}} h^{(k)}(D,V),$$

where

$$\begin{aligned} h^{(k)}(D,V) &= \Omega(D,V) - \Omega(D^{(k)}, V^{(k)})\\ &+ \nabla_D F(D^{(k)}, V^{(k)})^\top (D - D^{(k)}) + \gamma_k^{-1}\|D - D^{(k)}\|^2/2\\ &+ \nabla_V F(D^{(k)}, V^{(k)})^\top (V - V^{(k)}) + \gamma_k^{-1}\|V - V^{(k)}\|^2/2. \end{aligned}$$

Algorithm 1 SCADA

```
Require: Parameter $\delta \in ]0,1[$
  Set $D^{(0)} \sim \mathcal{N}(0_{P\times M}, 1_{P\times M})$ and $V^{(0)} = 0_{M\times N}$
  for $k = 0$ to $K-1$ do
    Provide rough estimate of $\gamma_k > 0$
    Proximal-gradient step
    $D^{(k+1/2)} = \mathrm{Proj}_{\mathcal{C}}(D^{(k)} - \gamma_k \nabla_D F(D^{(k)}, V^{(k)}))$
    $V^{(k+1/2)} = \mathrm{Soft}_{\gamma_k\lambda_1}(V^{(k)} - \gamma_k \nabla_V F(D^{(k)}, V^{(k)}))$
    Compute the difference
    $d_D^{(k)} = D^{(k+1/2)} - D^{(k)}$ & $d_V^{(k)} = V^{(k+1/2)} - V^{(k)}$
    Armijo-like backtracking loop
    $i_k = 0$ and $h_k = h^{(k)}(D^{(k+1/2)}, V^{(k+1/2)})$
    repeat
      $\tilde{D}^{(k)} = D^{(k)} + \delta^{i_k} d_D^{(k)}$ & $\tilde{V}^{(k)} = V^{(k)} + \delta^{i_k} d_V^{(k)}$
      $i_k = i_k + 1$
    until $\mathcal{L}(\tilde{D}^{(k)}, \tilde{V}^{(k)}) \leq \mathcal{L}(D^{(k)}, V^{(k)}) + \delta^{i_k} h_k$
    $D^{(k+1)} = \tilde{D}^{(k)}$ and $V^{(k+1)} = \tilde{V}^{(k)}$
  end for
  return Dictionary $D^{(K)}$, coding vector $V^{(K)}$
```

The overall step can be recast as

$$\begin{pmatrix} D^{(k+1/2)} \\ V^{(k+1/2)} \end{pmatrix} = \mathrm{prox}_{\gamma_k \Omega}\left(\begin{pmatrix} D^{(k)} \\ V^{(k)} \end{pmatrix} - \gamma_k \nabla F(D^{(k)}, V^{(k)})\right)$$

where the gradient is computed jointly over D and V. Note that since Ω is separable, it yields that

$$\begin{aligned} \mathrm{prox}_{\gamma_k\Omega}(D,V) &= \left(\mathrm{prox}_{\iota_{\mathcal{C}}}(D), \mathrm{prox}_{\gamma_k\lambda_1\|\cdot\|_1}(V)\right), \\ &= \left(\mathrm{Proj}_{\mathcal{C}}(D), \mathrm{Soft}_{\gamma_k\lambda_1}(V)\right), \end{aligned}$$

where $\mathrm{Soft}_{\gamma_k\lambda_1}(V)$ is the soft-thresholding operator [DDDM04]. The full scheme is sketched in Algorithm 1 while convergence guarantees are provided below.

Theorem 3.1 (Convergence [BLP+17]). *Let $\{D^{(k)}, V^{(k)}\}_{k\in\mathbb{N}}$ be the sequence of Algorithm 1. Then each limit point of $\{D^{(k)}, V^{(k)}\}_{k\in\mathbb{N}}$ is a stationary point of Problem 2.1 and $\{\mathcal{L}(D^{(k)}, V^{(k)})\}_{k\in\mathbb{N}}$ converges towards the objective value at the limit point. In addition, if $\mathcal{L}$ satisfies the Kurdyka-Łojasiewicz (KŁ) property at any point, then the sequence converges to a stationary point of Problem 2.1.*

Remark 3.1. Many functions met in neural networks are semi-algebraic or tame, and therefore satisfy the KŁ property (see, e.g., [ABS11, ZLLY19]). Since these "concepts" are stable under many operations, it is reasonable to assume that many DNN f are likely to satisfy the KŁ property and so does $\mathcal{L}$.

Once the dictionary is learned, one can attack a new example $x \sim \mu$ by solely learning the coding vector v. This amounts in solving (2) for fixed D through Algorithm 1 where the optimization steps over D have been omitted. In order to make sure that the adversarial example x' is a valid image, we additionally perform the projection $x' = \mathrm{Proj}_{[0,1]^P}\left(x + Dv^{(K)}\right)$.

4 Numerical experiments

In this section, the proposed approach, coined SCADA for *Sparse Coding of ADversarial Attacks*, is evaluated on numerical experiments.

Setting. The purpose of this experiment is to attack a LeNet [LBBH98] model f trained on the CIFAR-10 dataset and achieving a test accuracy of 62.8%. Although f is not smooth, we still apply our framework since it is unlikely that one lies at a discontinuity point during testing. To this regard, the dictionary D (with $M = 8$ atoms) is learned using a first subset $\mathcal{T}_1$ made of $N = 128$ images sampled from the test set. The adversary targets $\{t_i\}_{i=1}^N$ are chosen as the second most probable targets predicted by f. Then, a second subset $\mathcal{T}_2$ of 128 images is sampled from the test set in order to evaluate to what extent D is relevant to craft attacks to unseen examples x from $\mathcal{T}_2$. More precisely, $\mathcal{T}_2$ is used to learn coding vectors v so that $x + Dv$ is an adversarial example.

Illustration. For $(\lambda_1, \lambda_2) = (10^{-1}, 10^{-1})$ the training loss is reported in Figure 3 (left) whereas the learned adversarial dictionary is illustrated in Figure 1. Interestingly, we observe that the atoms are not purely random but unveil spatial patterns. Preliminary experiments, not reported here, suggest that these patterns not only depend on the dataset on which the model f has been trained but also on its architecture. Hence, the atoms of D highlight the most common flaws to fool f on CIFAR-10 images. We additionally report two SCADA attacks in Figure 2. As expected, the perturbation needed to attack each image only involves a few dictionary atoms. Using the proposed approach, one can understand the semantic used to fool the model. For instance, in the top example, the original image is attacked by a perturbation adding red and subtracting green to the bottom right area which happens to contains the frog. Learning D using the Pytorch implementation on a MacBook Pro 2,3 GHz Intel Core i9 with 8 cores required about 0.524 seconds per iteration. Once the dictionary D is learned, an attack is produced in 1.5 seconds.

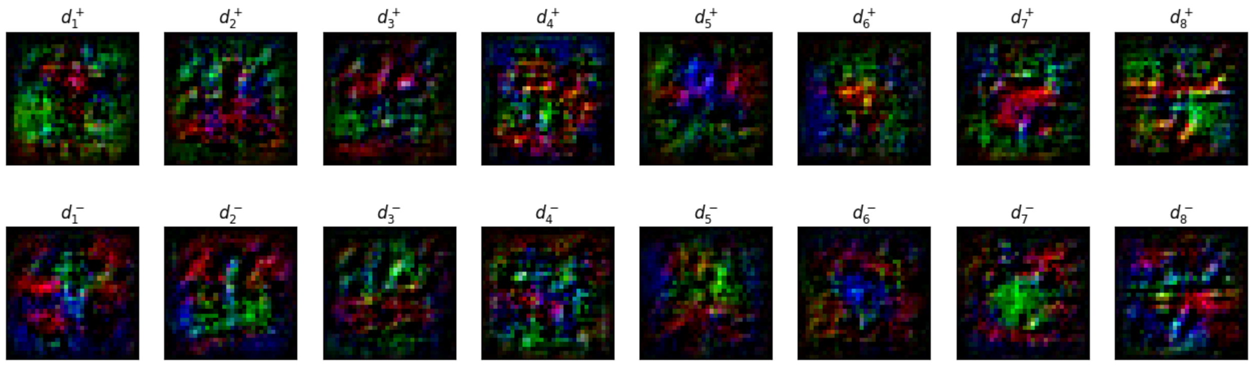

Figure 1: Dictionary of attacks for LeNet trained on CIFAR-10. Atoms are represented from left to right while their positive and negative parts are represented at the top and bottom, respectively. All atoms have been rescaled for display purposes.

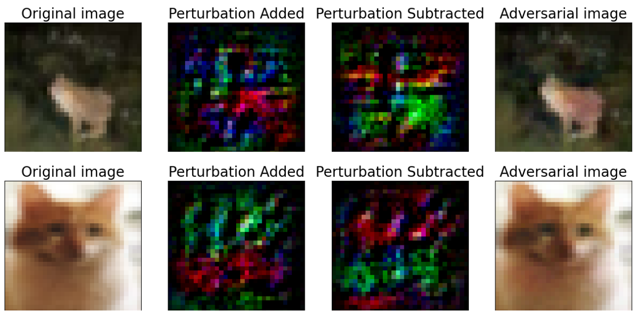

Figure 2: Two examples of proposed attacks. Top: $v = [0, 0.6269, 0, 0, 0, 0, 0, -0.7819]$ and t=*Deer*. Bottom: $v = [0, 0.3351, -0.5497, 0.021, 0, 0, 0, 0]$) and t=*Dog*.

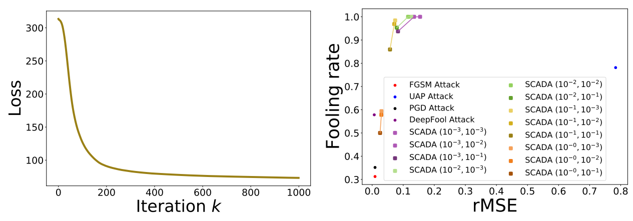

Figure 3: Left: training loss wrt iterations. Right: comparison against state-of-the-art attacks.

Impact of regularization parameters. Since the choice of (λ_1, λ_2) strongly affects the estimated attacks, we have examined their impacts by tuning them in $\{10^{-3}, 10^{-2}, 10^{-1}, 1\}$. The quality of the attacks is measured in terms of relative mean square error (rMSE), defined as $(1/|\mathcal{T}_2|)\sum_{i=1}^{|\mathcal{T}_2|} \|Dv_i\|^2/\|x_i\|^2$, and fooling rate. The former metric measures the proportion of noise added to the adversarial example while the latter metric is the percentage of adversarial examples fooling the classifier. Results, illustrated in Figure 3 (right), support that increasing λ_1 and decreasing λ_2 permit to increase the fooling rate at the cost of a higher rMSE. A good compromise is attained for $(\lambda_1, \lambda_2) = (10^{-1}, 10^{-3})$.

Comparisons with state-of-the-art. The same experiment is conducted for state-of-the-art attacks. Results, reported in Figure 3 (right), highlight two extremes. On the one hand, specific attacks (FGSM, PGD and DeepFool) achieve a small rMSE but fail to often fool the classifier. On the other hand, the universal attack UAP benefits from a higher fooling rate at the price of a high rMSE. By comparison, the proposed SCADA offers a good trade-off between the two ends of the spectrum. It is worth mentioning that, by varying the number M of atoms, one could join the two ends.

5 Conclusion

The present paper introduced SCADA, a dictionary learning framework for finding adversarial perturbations in the form of a sparse linear combination of shared dictionary elements. Once the dictionary is learned, it allows to craft an adversary perturbation to a new image by solely learning a few linear coefficients. Numerical experiments showed that SCADA offers a better trade-off between fooling rate and rMSE than state-of-the-art approaches while also benefiting from more interpretable attacks. Future works include an in-depth study for various dictionary constraints and neural network architectures, along with the design of a solver taking full advantage of the finite-sum nature of the optimization problem.

References

[ABS11] Hedy Attouch, Jérôme Bolte, and Benar Fux Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods. *Mathematical Programming*, 137(1-2):91–129, aug 2011.

[AM18] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430, 2018.

[BACM19] Rachel Blin, Samia Ainouz, Stéphane Canu, and Fabrice Meriaudeau. Road scenes analysis in adverse weather conditions by polarization-encoded images and adapted deep learning. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 27–32. IEEE, 2019.

[BLP$^+$17] Silvia Bonettini, Ignace Loris, Federica Porta, Marco Prato, and Simone Rebegoldi. On the convergence of a linesearch based proximal-gradient method for nonconvex optimization. *Inverse Problems*, 33(5):055005, mar 2017.

[CW17] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017.

[DDDM04] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 57(11):1413–1457, 2004.

[DHK13] Li Deng, Geoffrey Hinton, and Brian Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8599–8603. IEEE, 2013.

[FBI$^+$19] Samuel G. Finlayson, John D. Bowers, Joichi Ito, Jonathan L. Zittrain, Andrew L. Beam, and Isaac S. Kohane. Adversarial attacks on medical machine learning. *Science*, 363(6433):1287–1289, 2019.

[GSS15] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[KGB17] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.

[LBBH98] Yann Lecun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LF19] Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 10408–10418. Curran Associates, Inc., 2019.

[MDFFF17] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jul 2017.

[MFF16] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2574–2582. IEEE Computer Society, 2016.

[MLY17] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in bioinformatics*, 18(5):851–869, 2017.

[MMS$^+$19] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and

Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.

[MPS+09] Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis Bach. Supervised dictionary learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2009.

[Rak13] Alain Rakotomamonjy. Direct optimization of the dictionary learning problem. *IEEE Transactions on Signal Processing*, 61(22):5495–5506, 2013.

[Sra12] Suvrit Sra. Scalable nonconvex inexact proximal splitting. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[SV18] Kevin Scaman and Aladin Virmaux. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 3839–3848, 2018.

[SZS+14] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[YHPC18] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

[ZLLY19] Jinshan Zeng, Tim Tsz-Kit Lau, Shaobo Lin, and Yuan Yao. Global convergence of block coordinate descent in deep learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7313–7323. PMLR, 09–15 Jun 2019.