

**UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERIA
DEPARTAMENTO DE INGENIERIA INFORMÁTICA**



**ANÁLISIS DE LA EFICIENCIA DEL ENTRENAMIENTO DE REDES
NEURONALES PROFUNDAS BASADO EN SIMULATED ANNEALING**

Felipe Alberto Reyes González

Profesor Guía: Victor Parada

Tesis de grado presentada en conformidad a los
requisitos para obtener el grado de Magíster en
Ingeniería Informática

Santiago, Chile

2017

© Felipe Alberto Reyes González- 2017



• Algunos derechos reservados. Esta obra está bajo una Licencia Creative Commons Atribución-Chile 3.0. Sus condiciones de uso pueden ser revisadas en:

<http://creativecommons.org/licenses/by/3.0/cl/>.

TABLA DE CONTENIDO

1	Introducción	1
1.1	Antecedentes y motivación	1
1.2	Descripción del problema	3
1.3	Solución propuesta	4
1.3.1	Características de la solución	4
1.3.2	Propósito de la solución	4
1.4	Objetivos y alcances del proyecto	4
1.4.1	Objetivo general	5
1.4.2	Objetivos específicos	5
1.4.3	Alcances	5
1.5	Metodología y herramientas utilizadas	6
1.5.1	Metodología de trabajo	6
1.5.2	Herramientas de desarrollo	6
2	Aspectos teóricos y revisión de la literatura	8
2.1	Aspectos teóricos	8
2.2	Revisión de la literatura	8
	Referencias	9

ÍNDICE DE TABLAS

Tabla 1.1 Especificaciones del equipo 7

ÍNDICE DE ILUSTRACIONES

INTRODUCCIÓN

ANTECEDENTES Y MOTIVACIÓN

Las redes neuronales artificiales (*Artificial Neural Networks*, NN) han sido testigos de un renacimiento en el campo de las máquinas de aprendizaje (*Machine learning*, ML) con el surgimiento del *aprendizaje profundo* (Bengio et al., 2006; Hinton et al., 2006; Le et al., 2012; Ranzato et al., 2007). Las principales ideas detrás del nuevo enfoque abarcan una gama de algoritmos (Bengio y LeCun, 2007; Hinton et al., 2006), pero un principio unificador es que una NN con múltiples capas ocultas (que lo hacen profundo) puede codificar características cada vez más complejas en sus capas superior. Las NN fueron comunmente entrenadas a través del algoritmo de retropropagación (Rumelhart et al., 1986b), que aplica el gradiente estocástico descendente (*Stochastic gradient descent*, SGD) o una de sus variantes a los pesos de la NN para reducir el error total, y no fue hasta el año 2006 que fue difundido que el método del gradiente disminuiría el valor del gradiente en las redes profundas. Sin embargo, los descubrimientos en los últimos años han demostrado que, con suficientes datos de entrenamiento y poder de procesamiento, el método de retropropagación y SGD resultan ser eficaces en la optimización de una NN masiva con millones de conexiones y muchas capas (Cireşan et al., 2012; He et al., 2015; Le et al., 2012). Esta realización ha llevado a registros sustantivos que se rompen en muchas áreas de las ML a través de la aplicación de la retropropagación en el aprendizaje profundo (Cireşan et al., 2012; He et al., 2015; Le et al., 2012), incluyendo el aprendizaje no supervisado (Bengio, 2009).

La forma más común de aprendizaje automático, profundo o no, es el aprendizaje supervisado. Imagínese que queremos construir un sistema que pueda clasificar las imágenes como conteniendo, digamos, una casa, un automóvil, una persona o una mascota. Primero recopilamos un gran conjunto de datos de imágenes de casas, autos, personas y mascotas, cada uno etiquetado con su categoría. Durante el entrenamiento, la máquina muestra una imagen y produce una salida en forma de un vector de puntuaciones, uno para cada categoría. Queremos que la categoría deseada tenga la puntuación más alta de todas las categorías, pero es poco probable que suceda antes del entrenamiento. Se calcula una función objetivo que mide el error (o distancia) entre las puntuaciones de salida y el patrón de puntuaciones deseado. La máquina modifica entonces sus parámetros ajustables internos para reducir este error. Estos parámetros ajustables, a menudo llamados pesos, son números reales que pueden ser vistos como "perillas" que definen la función de entrada-salida de la máquina. En un típico sistema de aprendizaje profundo, puede haber cien-

tos de millones de estos pesos ajustables y Cientos de millones de ejemplos etiquetados con los que entrenar la máquina.

Las NN han sido ampliamente estudiadas y ampliamente utilizadas en muchas aplicaciones de la inteligencia artificial. El problema durante el proceso de aprendizaje de las NN es descrito como un problema de minimización de una función de error, la que depende de los pesos que conforman la red (Rumelhart et al., 1986a). Este problema de optimización tiene la desventaja de ser no lineal, no convexo, además de tener mas de un mínimo local. Para solventar este problema se han desarrollado diversos algoritmos (Grippo, 1994; Jacobs, 1988; V. P. Plagianakos et al., 2002; Rumelhart et al., 1986b; V. Plagianakos et al., 1998) y su rendimiento varía según el problema a resolver.

El enfoque clásico para el entrenamiento de las NN es la aplicación de algoritmos basados en el gradiente como la retropropagación (Rumelhart et al., 1986b). El algoritmo de retropropagación busca minimizar la función de error mediante la dirección de descenso más pronunciada. Aunque la función de error disminuye rápidamente en la dirección del gradiente negativo, la retropropagación es generalmente ineficiente y poco fiable (Gori y Tesi, 1992) debido a la superficie de error. Además, su rendimiento se ve afectado por parámetros que deben ser especificados por el usuario, pues no existe una base teórica para escogerlos (Nguyen y Widrow, 1990). Dichos parámetros tienen una importancia crucial en el buen funcionamiento del algoritmo, por lo que el diseñador está obligado a seleccionar parámetros como los pesos iniciales de la NN, la topología de la red y la tasa de aprendizaje. En diversas investigaciones (Cauchy, 1847; Grippo, 1994; V. Plagianakos et al., 1998; V. P. Plagianakos et al., 2002) ha quedado demostrado que pequeñas modificaciones en estos valores influyen en el rendimiento de la NN.

Para proporcionar una convergencia más rápida y estable se han desarrollado diversas variaciones y alternativas a la retropropagación. Algunos de estos métodos son la adaptación de un término de momento (Jacobs, 1988; Rumelhart et al., 1986b) o de una tasa variable de aprendizaje (Jacobs, 1988; Vogl et al., 1988). Magoulas, Vrahatis, y Androulakis (1997); V. Plagianakos et al. (1998) propusieron dos técnicas para evaluar en forma dinámica la tasa de aprendizaje sin el uso de alguna heurística o alguna función adicional y las evaluaciones de gradiente. El primero se basó en el algoritmo de Barzilai y Borwein (Barzilai y Borwein, 1988) que adapta la tasa de aprendizaje sin evaluar la matriz Hessiana; mientras que el segundo utiliza estimaciones de la constante de Lipschitz, explotando la información local de la superficie de error y los pesos posteriores (Magoulas et al., 1997).

Se han sugerido métodos de segundo orden para mejorar la eficiencia del proceso de minimización del error. Algunos de los métodos utilizados son el del gradiente conjugado (Fletcher y

Reeves, 1964; Hestenes y Stiefel, 1952; Polak E., 1969) y el quasi-Newton (Huang, 1970; Nocedal y Wright, 2006). Los métodos del gradiente conjugado utiliza una combinación lineal de la dirección de búsqueda anterior y el gradiente actual lo que produce una convergencia generalmente más rápida, es adecuado para redes neuronales de gran escala debido a su simplicidad, sus propiedades de convergencia y la poca memoria que requiere. En la literatura se encuentran diversos métodos basados en el gradiente conjugado (Birgin y Martínez, 2001; Møller, 1993) que han sido utilizados para la construcción de NN en varias aplicaciones (Charalambous, 1992; Peng y Magoulas, 2007; Sotiropoulos et al., 2002). Los métodos quasi-Newton se consideran como los algoritmos más sofisticados para el entrenando rápido de una NN. Definen la dirección de búsqueda mediante una aproximación de la matriz Hessiana, requiriendo información adicional. Se han propuesto soluciones para ajustar la aproximación Hessiana mediante la introducción de distintas estrategias (Al-Baali, 1998; Nocedal y Yuan, 1993; S. Oren, 1972; S. S. Oren y Luenberger, 1974; Yin y Du, 2007). Estas estrategias combinadas con búsquedas lineales no monótonas permitieron definir una convergencia superlineal y global (Yin y Du, 2007), mejorando significativamente el rendimiento de los métodos originales.

El aprendizaje profundo (*Deep learning*, DL) se refiere a una nueva clase de métodos de las máquinas de aprendizaje (*Machine learning*, ML), donde el proceso de información de muchas capas en arquitecturas jerárquicas se puede utilizar para clasificar el patrón y el aprendizaje de la característica. Esta técnica se encuentra en las intersecciones entre las áreas de investigación en el modelo gráfico, optimización, procesamiento de señales, reconocimiento de patrones y redes neuronales. Tres razones que explican la popularidad del DL es que aumenta drásticamente las capacidades de procesamiento de chips, reduce significativamente el costo de hardware de computación y los recientes avances en la investigación de ML [1].

En general, las técnicas del DL pueden clasificarse en modelos discriminativos profundos y modelos generativos [2]. Ejemplos de modelos discriminativos son redes neuronales profundas (*Deep neural networks*, DNN), redes neuronales recurrentes (*Recurrent neural networks*, RNN) y redes neuronales convolucionales (*Convolutional neural networks*, CNN). Por otro lado, los modelos generativos, por ejemplo, son máquinas de Boltzmann restringidas (*Restricted Boltzmann machine*, RBMs), redes de creencias profundas (*Deep belief networks*, DBN), autocodificadores regularizados y máquinas profundas de Boltzmann (DBMs). Entre estas técnicas, CNNs es el foco de este documento. CNNs es un profundo aprendizaje supervisado, mientras que este modelo suele ser eficiente para entrenar y probar, flexible para construir, y adecuado para el aprendizaje de extremo a extremo del sistema complejo [2].

DESCRIPCIÓN DEL PROBLEMA

La retropropagación basa su funcionamiento en multiplicaciones sucesivas basadas en el error para poder calcular los gradientes, y a medida que el error se propaga hacia la capa de entrada de la red el gradiente comienza a disminuir su valor por cada capa que atraviesa. Esto significa que el gradiente disminuirá de manera exponencial, lo que representa un problema para redes profundas, ya que las capas mas cercanas a la capa de entrada necesitarán más tiempo para ser entrenadas.

El método de aprendizaje basado en simulated annealing permite la actualización de los pesos de la red sin mermar la capacidad de adaptación de los pesos. El método supone una alternativa efectiva a los métodos tradicionales de aprendizaje para la convergencia de los métodos debido a la independencia que otorga a la actualización de los pesos de las distintas capas..

SOLUCIÓN PROPUESTA

Características de la solución

Mediante el uso de el algoritmo *simulated annealing* se busca analizar la eficiencia que la NN alcanza en una red neuronal profunda frente a otros métodos de aprendizaje tales como SGD y RMSPROP.

Propósito de la solución

El propósito de la solución es aportar en el campo de las redes neuronales y la clasificación de datos, proporcionando un análisis comparativo de la convergencia de distintas redes.

OBJETIVOS Y ALCANCES DEL PROYECTO

En ésta sección se presenta el objetivo general, los objetivos específicos además del alcance y limitaciones de la presenta investigación.

Objetivo general

Evaluar el desempeño del algoritmo *simulated annealing* y su efecto sobre el entrenamiento de redes neuronales profundas en comparación con otros métodos.

Objetivos específicos

Los objetivos establecidos para el presente trabajo son descritos a continuación

1. Definir las reglas de aprendizaje a comparar.
2. Construir los conjuntos de datos de entrada y salida a analizar.
3. Establecer los parámetros de las redes neuronales para la experimentación.
4. Establecer los algoritmos de aprendizaje a comparar.
5. Entrenar las redes con los distintos conjuntos de datos.
6. Establecer las conclusiones del trabajo.

Alcances

1. Se analizará la misma arquitectura con diferentes reglas de aprendizaje.
2. Los conjunto de datos para el entrenamiento a utilizar son los propuestos en (Morse y Stanley, 2016).

METODOLOGÍA Y HERRAMIENTAS UTILIZADAS

Metodología de trabajo

Considerando el aspecto investigativo del trabajo, se considera la utilización del método científico. Entre las actividades que componen la metodología, Sampieri (2006) describe los siguientes pasos para desarrollar una investigación:

- Formulación de la hipótesis: Las redes neuronales que adolecen del desvanecimiento del gradiente se ven beneficiadas por el uso del algoritmo *simulated annealing* en la convergencia.
- Marco teórico: Una revisión de la literatura donde se aborda el problema planteado, para situarse en el contexto actual de los problemas. Se describirán redes neuronales que buscan solucionar el mismo problema.
- Diseño de la solución: Se deberá diseñar el experimento para generar los datos que permitan sustentar las comparaciones entre las distintas redes.
- Análisis y verificación de los resultados: Los resultados se analizarán considerando los valores de convergencia de los distintos métodos.
- Presentación de los resultados: Se presentarán tablas que describan los resultados obtenidos y que se consideren pertinentes.
- Conclusiones obtenidas en el desarrollo de la investigación.

Herramientas de desarrollo

Para el desarrollo y ejecución de los experimentos se utilizará un equipo con las siguientes características

Sistema Operativo	Solus 2017.04.18.0 64-bit
Procesador	Intel® Core™i5-2450M CPU @ 2.50GHz x 4
RAM	7.7Gb
Gráficos	Intel® Sandybridge Mobile
Almacenamiento	935.6 GB

Tabla 1.1: Especificaciones del equipo

El software que se utilizará es:

- Lenguaje de programación: Python.
- Sistema de redes neuronales: Keras API (Chollet, 2015).
- Herramienta ofimática: \LaTeX .

ASPECTOS TEÓRICOS Y REVISIÓN DE LA LITERATURA

ASPECTOS TEÓRICOS

REVISIÓN DE LA LITERATURA

REFERENCIAS

- Al-Baali, M. (1998). Numerical experience with a class of self-scaling quasi-newton algorithms. *Journal of Optimization Theory and Applications*, 96(3), 533–553. doi: 10.1023/A:1022608410710
- Barzilai, J., y Borwein, J. M. (1988). Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1), 141. doi: 10.1093/imanum/8.1.141
- Bengio, Y. (2009, enero). Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1), 1–127. Descargado de <http://dx.doi.org/10.1561/22000000006> doi: 10.1561/22000000006
- Bengio, Y., Lamblin, P., Popovici, D., y Larochelle, H. (2006). Greedy layer-wise training of deep networks. En *Proceedings of the 19th international conference on neural information processing systems* (pp. 153–160). Cambridge, MA, USA: MIT Press. Descargado de <http://dl.acm.org/citation.cfm?id=2976456.2976476>
- Bengio, Y., y LeCun, Y. (2007). Scaling learning algorithms towards AI. En L. Bottou, O. Chapelle, D. DeCoste, y J. Weston (Eds.), *Large-scale kernel machines*. MIT Press. Descargado de <http://yann.lecun.com/exdb/publis/pdf/bengio-lecun-07.pdf>
- Birgin, E. G., y Martínez, J. M. (2001). A spectral conjugate gradient method for unconstrained optimization. *Applied Mathematics and Optimization*, 43(2), 117–128. Descargado de <http://dx.doi.org/10.1007/s00245-001-0003-0> doi: 10.1007/s00245-001-0003-0
- Cauchy, A.-L. (1847, 18 de octubre). Méthode générale pour la résolution des systèmes d'équations simultanées. *Compte Rendu des S'eances de L'Acad'emie des Sciences XXV, S'erie A*(25), 536–538.
- Charalambous, C. (1992, June). Conjugate gradient algorithm for efficient training of artificial neural networks. *IEE Proceedings G - Circuits, Devices and Systems*, 139(3), 301-310. doi: 10.1049/ip-g-2.1992.0050
- Chollet, F. (2015). *Keras*. <https://github.com/fchollet/keras>. GitHub.
- Cireřan, D., Meier, U., Masci, J., y Schmidhuber, J. (2012). Multi-column deep neural network for traffic sign classification. *Neural Networks*, 32, 333 - 338. Descargado de <http://www.sciencedirect.com/science/article/pii/S0893608012000524> (Selected Papers from {IJCNN} 2011) doi: <https://doi.org/10.1016/j.neunet.2012.02.023>

- Fletcher, R., y Reeves, C. M. (1964, 1 de febrero). Function minimization by conjugate gradients. *The Computer Journal*, 7(2), 149–154. Descargado de <http://dx.doi.org/10.1093/comjnl/7.2.149> doi: 10.1093/comjnl/7.2.149
- Gori, M., y Tesi, A. (1992, Jan). On the problem of local minima in backpropagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1), 76-86. doi: 10.1109/34.107014
- Grippo, L. (1994). A class of unconstrained minimization methods for neural network training. *Optimization Methods and Software*, 4(2), 135-150. doi: 10.1080/10556789408805583
- He, K., Zhang, X., Ren, S., y Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385. Descargado de <http://arxiv.org/abs/1512.03385>
- Hestenes, M. R., y Stiefel, E. (1952). Methods of conjugate gradients for solving linear systems. *Journal of research of the National Bureau of Standards*, 49, 409–436.
- Hinton, G. E., Osindero, S., y Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7), 1527-1554. Descargado de <http://dx.doi.org/10.1162/neco.2006.18.7.1527> (PMID: 16764513) doi: 10.1162/neco.2006.18.7.1527
- Huang, H. Y. (1970). Unified approach to quadratically convergent algorithms for function minimization. *Journal of Optimization Theory and Applications*, 5(6), 405–423. Descargado de <http://dx.doi.org/10.1007/BF00927440> doi: 10.1007/BF00927440
- Jacobs, R. A. (1988). Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1(4), 295 - 307. doi: [https://doi.org/10.1016/0893-6080\(88\)90003-2](https://doi.org/10.1016/0893-6080(88)90003-2)
- Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., ... Ng, A. Y. (2012). Building high-level features using large scale unsupervised learning. En *Proceedings of the 29th international conference on machine learning* (pp. 507–514). USA: Omnipress. Descargado de <http://dl.acm.org/citation.cfm?id=3042573.3042641>
- Magoulas, G. D., Vrahatis, M. N., y Androulakis, G. S. (1997). Effective backpropagation training with variable stepsize. *Neural Networks*, 10(1), 69 - 82. Descargado de <http://www.sciencedirect.com/science/article/pii/S0893608096000524> doi: [https://doi.org/10.1016/S0893-6080\(96\)00052-4](https://doi.org/10.1016/S0893-6080(96)00052-4)
- Morse, G., y Stanley, K. O. (2016). Simple evolutionary optimization can rival stochastic gradient descent in neural networks. En *Proceedings of the genetic and evolutionary computation conference 2016* (pp. 477–484). New York, NY, USA: ACM. Descargado de <http://doi.acm.org/10.1145/2908812.2908916> doi: 10.1145/2908812.2908916

- Møller, M. F. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6(4), 525 - 533. Descargado de <http://www.sciencedirect.com/science/article/pii/S0893608005800565> doi: [https://doi.org/10.1016/S0893-6080\(05\)80056-5](https://doi.org/10.1016/S0893-6080(05)80056-5)
- Nguyen, D., y Widrow, B. (1990, June). Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. En *1990 ijcnn international joint conference on neural networks* (p. 21-26 vol.3). doi: 10.1109/IJCNN.1990.137819
- Nocedal, J., y Wright, S. (2006). *Numerical optimization (springer series in operations research and financial engineering)*. Springer.
- Nocedal, J., y Yuan, Y.-x. (1993). Analysis of a self-scaling quasi-newton method. *Mathematical Programming*, 61(1), 19–37. doi: 10.1007/BF01582136
- Oren, S. (1972). *Self-scaling variable metric algorithms for unconstrained minimization*. Department of Engineering-Economic Systems, Stanford University.
- Oren, S. S., y Luenberger, D. G. (1974). Self-scaling variable metric (ssvm) algorithms. part i: Criteria and sufficient conditions for scaling a class of algorithms. *Management Science*, 20(5), 845-862. Descargado de <http://www.jstor.org/stable/2630094>
- Peng, C. C., y Magoulas, G. D. (2007, Oct). Adaptive nonmonotone conjugate gradient training algorithm for recurrent neural networks. En *19th ieee international conference on tools with artificial intelligence(ictai 2007)* (Vol. 2, p. 374-381). doi: 10.1109/ICTAI.2007.126
- Plagianakos, V., Sotiropoulos, D., y Vrahatis, M. (1998). Automatic adaptation of learning rate for backpropagation neural networks. *Recent Advances in Circuits and Systems*, 337.
- Plagianakos, V. P., Magoulas, G. D., y Vrahatis, M. N. (2002, Nov). Deterministic nonmonotone strategies for effective training of multilayer perceptrons. *IEEE Transactions on Neural Networks*, 13(6), 1268-1284. doi: 10.1109/TNN.2002.804225
- Polak E., R. G. (1969). Note sur la convergence de méthodes de directions conjuguées. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, 3(R1), 35-43. Descargado de <http://eudml.org/doc/193115>
- Ranzato, M., Ian Boureau, Y., y Cun, Y. L. (2007). Sparse feature learning for deep belief networks. En J. Platt, D. Koller, Y. Singer, y S. Roweis (Eds.), *Advances in neural information processing systems 20* (pp. 1185–1192). Cambridge, MA: MIT Press. Descargado de http://books.nips.cc/papers/files/nips20/NIPS2007_1118.pdf
- Rumelhart, D. E., Hinton, G. E., y Williams, R. J. (1986a). Learning representations by back-propagating errors. *Nature*, 323, 533–536.

Rumelhart, D. E., Hinton, G. E., y Williams, R. J. (1986b). Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. En D. E. Rumelhart, J. L. McClelland, y C. PDP Research Group (Eds.), (pp. 318–362). Cambridge, MA, USA: MIT Press. Descargado de <http://dl.acm.org/citation.cfm?id=104279.104293>

Sampieri, R. (2006). *Metodología de la investigación*. México: McGraw Hill.

Sotiropoulos, D., Kostopoulos, A., y Grapsa, T. (2002). A spectral version of perry's conjugate gradient method for neural network training. En *Proceedings of 4th gracm congress on computational mechanics* (Vol. 1, pp. 291–298).

Vogl, T. P., Mangis, J. K., Rigler, A. K., Zink, W. T., y Alkon, D. L. (1988). Accelerating the convergence of the back-propagation method. *Biological Cybernetics*, 59(4), 257–263. doi: 10.1007/BF00332914

Yin, H. X., y Du, D. L. (2007). The global convergence of self-scaling bfgs algorithm with nonmonotone line search for unconstrained nonconvex optimization problems. *Acta Mathematica Sinica, English Series*, 23(7), 1233–1240. doi: 10.1007/s10114-005-0837-5