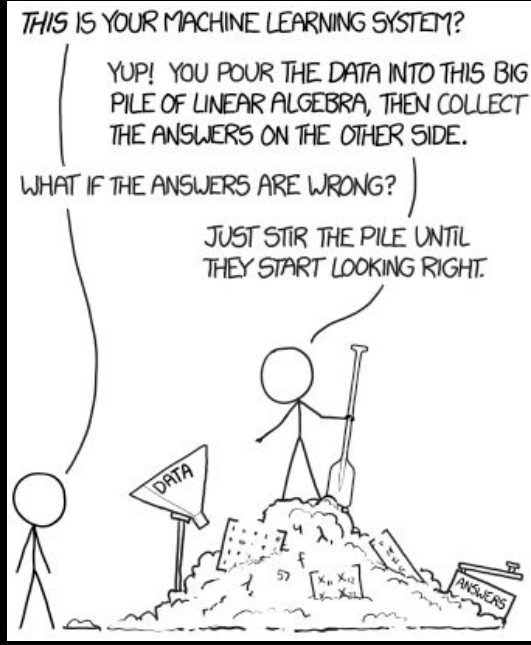
Regularização

Paulo Ricardo Lisboa de Almeida



https://xkcd.com/1838

DSBD

UFPR

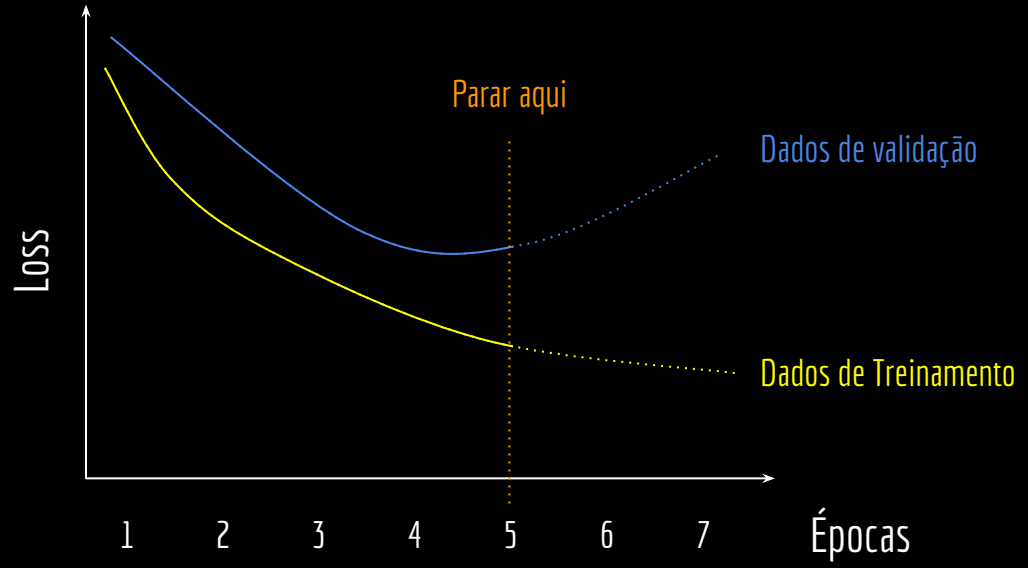
Regularização

“... como criar um algoritmo que funcionará bem não só com os dados de treinamento, mas também com novos dados de entrada (dados de teste). Muitas estratégias usadas no aprendizado de máquina são explicitamente criadas com o objetivo de reduzir o erro no teste, mesmo que isso custe um aumento do erro no treinamento. Essas estratégias são coletivamente conhecidas como regularização ...” (Goodfellow, Bengio, Courville; 2016).

Early Stopping

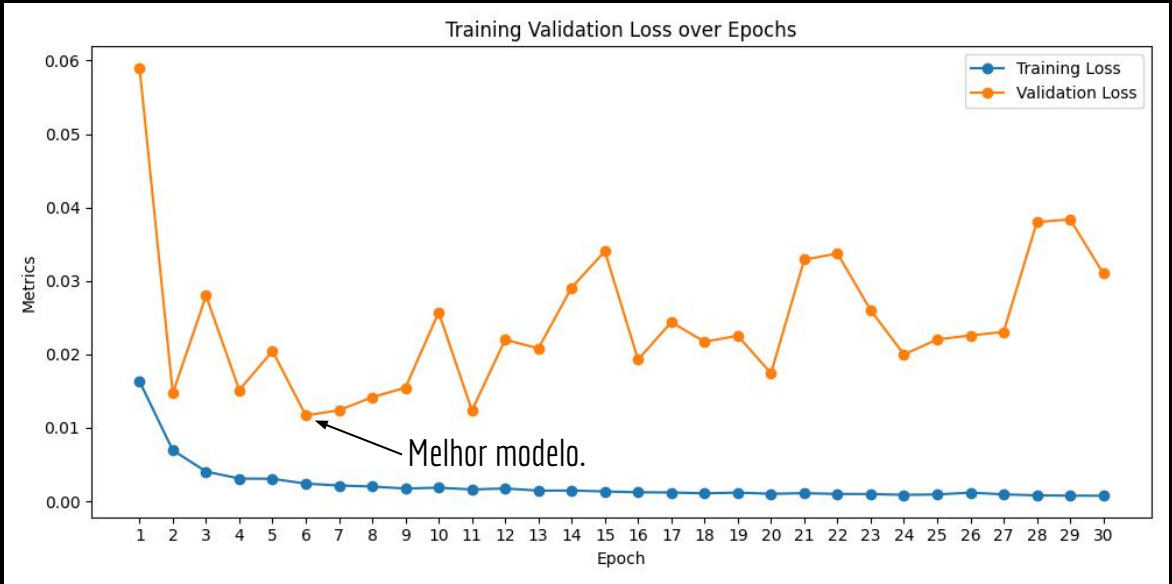
A técnica de usar um dataset de validação para escolher o modelo na melhor época é uma técnica de regularização.

Com Early Stopping, continuamos o treinamento da rede até que o erro na validação comece a aumentar.



No mundo real ...

Devido a vários fatores, como randomização dos dados de validação e mudanças na taxa de aprendizado, o Loss na validação pode não decrescer de forma monotônica.

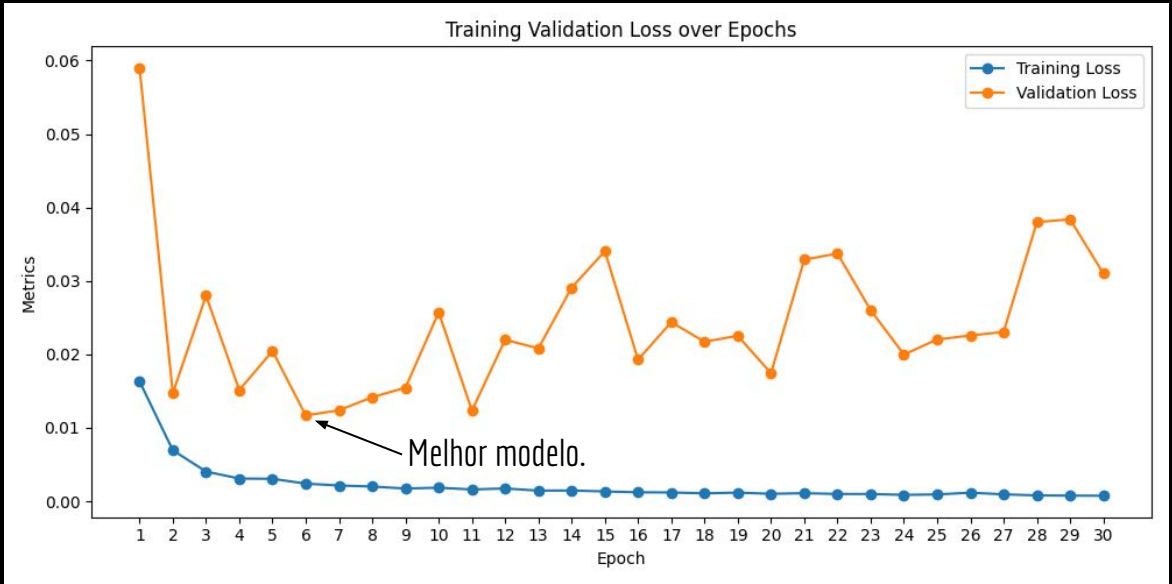


Dado do treinamento de um **modelo real** feito pela equipe do DSBD.

No mundo real ...

Uma prática comum é definir uma variável de “paciência”.

O modelo tem *x* épocas para melhorar o *loss*, caso contrário, o treinamento será finalizado.



Dado do treinamento de um **modelo real** feito pela equipe do DSBD.

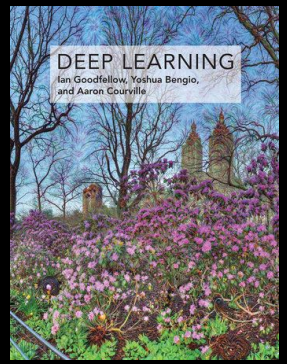
Early Stopping

Exemplo de função de Early Stopping pronta no Keras:

```
keras.callbacks.EarlyStopping(
    monitor="val_loss",
    min_delta=0,
    patience=0,
    verbose=0,
    mode="auto",
    baseline=None,
    restore_best_weights=False,
    start_from_epoch=0,
)
```

Veja o algoritmo em:

Goodfellow, I., Bengio, Y., Courville, A. Deep Learning. 2016.



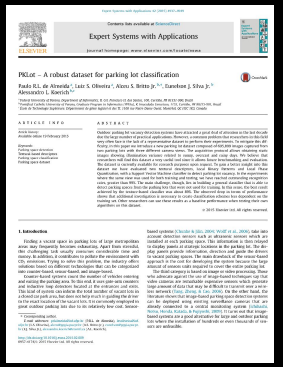
Data Augmentation

Ideia: aumentar o tamanho do dataset de treinamento artificialmente, para incluir variações de imagens esperadas no mundo real.

Vantagens:

- Relativamente simples de se implementar.
- “Ensinar” a rede as possíveis variações que podem existir no mundo real, mesmo que elas não existam diretamente no dataset de treinamento.
- Evita problemas quando temos poucas amostras de treinamento.
    - Veja overfitting e fenômeno de Runge da aula passada.

Exemplo



Expert Systems with Applications

Contents lists available at ScienceDirect

Journal homepage: www.elsevier.com/locate/eswa

PKLot – A robust dataset for parking lot classification

Paulo R.L. de Almeida[a], Luiz S. Oliveira[a], Alceu S. Britto Jr.[b,*], Eunelson J. Silva Jr.[b], Alessandro L. Koerich[c]

ARTICLE INFO

ABSTRACT

1. Introduction

Imagens do Dataset PKLot:
web.inf.ufpr.br/vri/databases/parking-lot-database

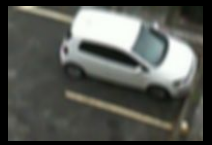
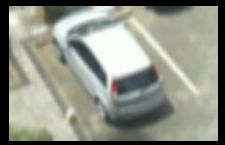
www.sciencedirect.com/science/article/abs/pii/S0957417415001086


Original


Horizontal Flip


Center Crop


Color Jitter

Gaussian Blur

Faça você mesmo

Execute os exemplos de Data Augmentation disponibilizados no Google Colab.

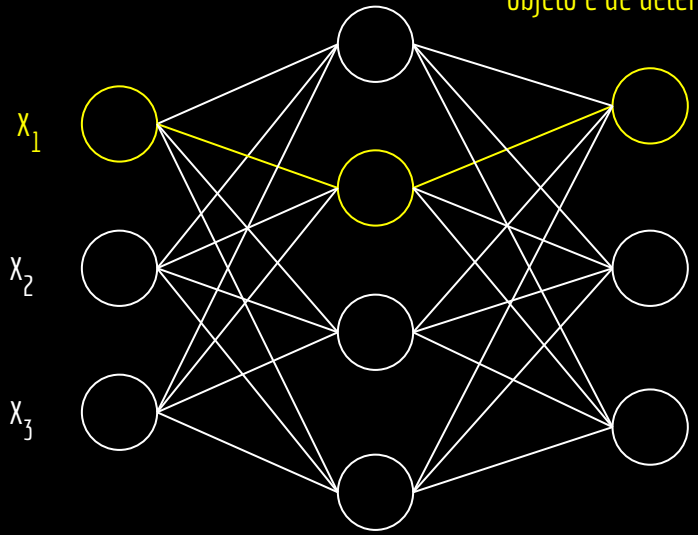
Carregue imagens diferentes.

Aplique augmentations disponíveis em https://pytorch.org/vision/stable/transforms.html

Dropout

Durante o treinamento a rede pode entrar em *overfit* devido a ruídos estatísticos no dado de treinamento.

Exemplo: por um pixel morto em x_1 a rede pode "aprender" que todo objeto é de determinada classe.



Dropout

Além disso, dados M neurônios de entrada e escondidos (*hidden*), existem 2^M combinações de arquiteturas de redes possíveis.

Como escolher uma?

Dropout

Dropout – aproximar os 2^M modelos através de uma espécie de média.

Procedimento: omitir nodos aleatoriamente da rede neural a cada amostra (ou batch) de treinamento.

Journal of Machine Learning Research 15 (2014) 1929-1958 Submitted 11/13; Published 6/14

Dropout: A Simple Way to Prevent Neural Networks from Overfitting

Nitish Srivastava NITISH@CS.TORONTO.EDU
Geoffrey Hinton HINTON@CS.TORONTO.EDU
Alex Krizhevsky KRIZ@CS.TORONTO.EDU
Ilya Sutskever ILYA@CS.TORONTO.EDU
Ruslan Salakhutdinov RSALAKHU@CS.TORONTO.EDU
Department of Computer Science
University of Toronto
10 Kings College Road, Rm 3302
Toronto, Ontario, M5S 3G4, Canada.

Editor: Yoshua Bengio

Abstract

Deep neural nets with a large number of parameters are very powerful machine learning systems. However, overfitting is a serious problem in such networks. Large networks are also slow to use, making it difficult to deal with overfitting by combining the predictions of many different large neural nets at test time. Dropout is a technique for addressing this problem. The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents units from co-adapting too much. During training, dropout samples from an exponential number of different "thinned" networks. At test time, it is easy to approximate the effect of averaging the predictions of all these thinned networks by simply using a single unthinned network that has smaller weights. This significantly reduces overfitting and gives major improvements over other regularization methods. We show that dropout improves the performance of neural networks on supervised learning tasks in vision, speech recognition, document classification and computational biology, obtaining state-of-the-art results on many benchmark data sets.

Keywords: neural networks, regularization, model combination, deep learning

1. Introduction

Deep neural networks contain multiple non-linear hidden layers and this makes them very expressive models that can learn very complicated relationships between their inputs and outputs. With limited training data, however, many of these complicated relationships will be the result of sampling noise, so they will exist in the training set but not in real test data even if it is drawn from the same distribution. This leads to overfitting and many methods have been developed for reducing it. These include stopping the training as soon as performance on a validation set starts to get worse, introducing weight penalties of various kinds such as L1 and L2 regularization and soft weight sharing (Nowlan and Hinton, 1992).

With unlimited computation, the best way to "regularize" a fixed-sized model is to average the predictions of all possible settings of the parameters, weighting each setting by

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*. 2014.

Dropout

Dropout – aproximar os 2^M modelos através de uma espécie de média.

Procedimento: omitir nodos aleatoriamente da rede neural a cada amostra (ou batch) de treinamento.

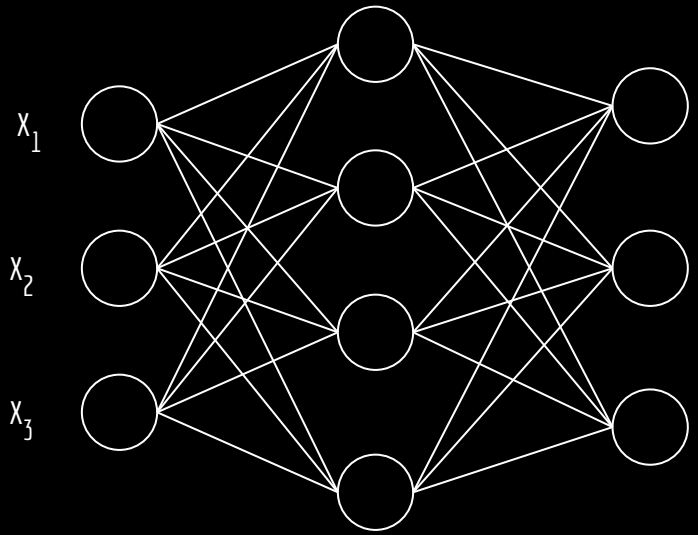
Omitir um nodo é o mesmo que multiplicar a função de ativação por **zero**.

Uma forma eficiente de se implementar é criar uma matriz de máscara $R_{il} \in \{0, 1\}$,onde *l* é a camada, e *i* é o índice do neurônio.

- Os valores da matriz são escolhidos aleatoriamente a cada amostra ou batch de treinamento.
- Os valores multiplicam as saídas das funções de ativação.

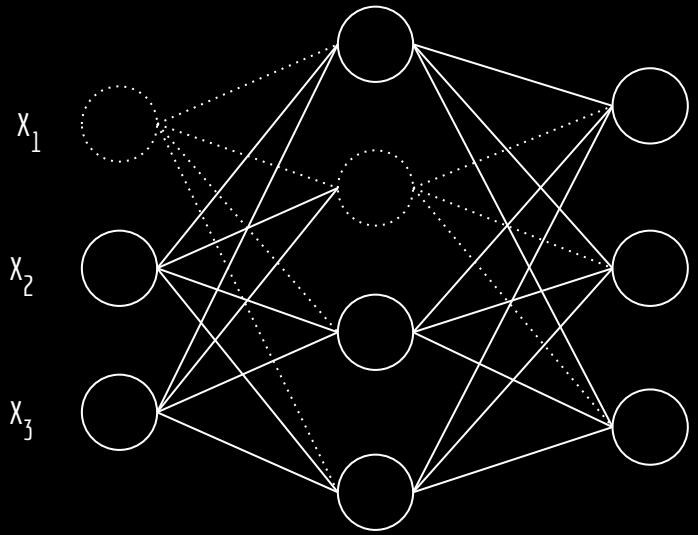
Dropout - Exemplo

Rede Original.



Dropout - Exemplo

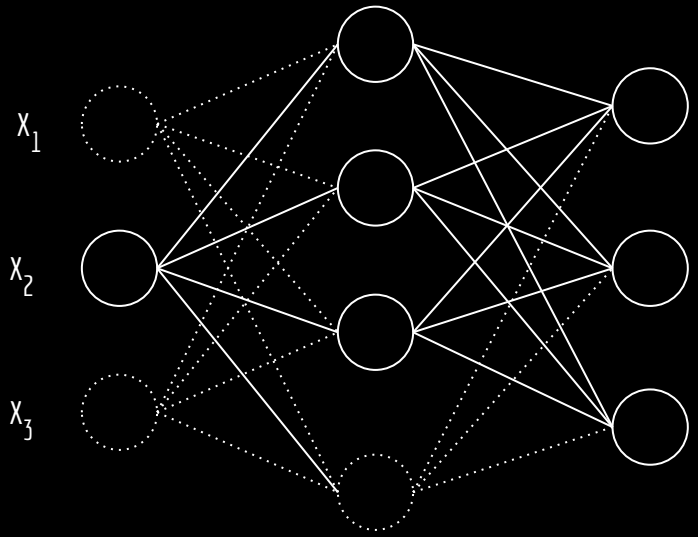
Primeiro Batch de treinamento.



$$R = \begin{vmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 1 \\ x & 1 \end{vmatrix}$$

Dropout - Exemplo

Segundo Batch de treinamento.



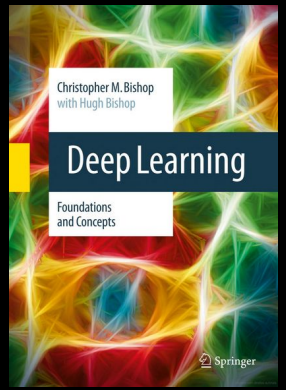
$$R = \begin{vmatrix} 0 & 1 \\ 1 & 1 \\ 0 & 1 \\ x & 0 \end{vmatrix}$$

Dropout

Considerando *p* como a probabilidade de determinado elemento da matriz de máscara ser 1 a cada iteração, valores típicos de *p* são:

- Camadas escondidas: *p = 0,5*
- Camada de entrada: *p = 0,8*

Bishop, C. M., Bishop, H. Deep Learning: Foundations and Concepts. 2023.



Dropout - Predições

Depois do treinamento, antes de realizar as predições precisamos ajustar a escala dos pesos.

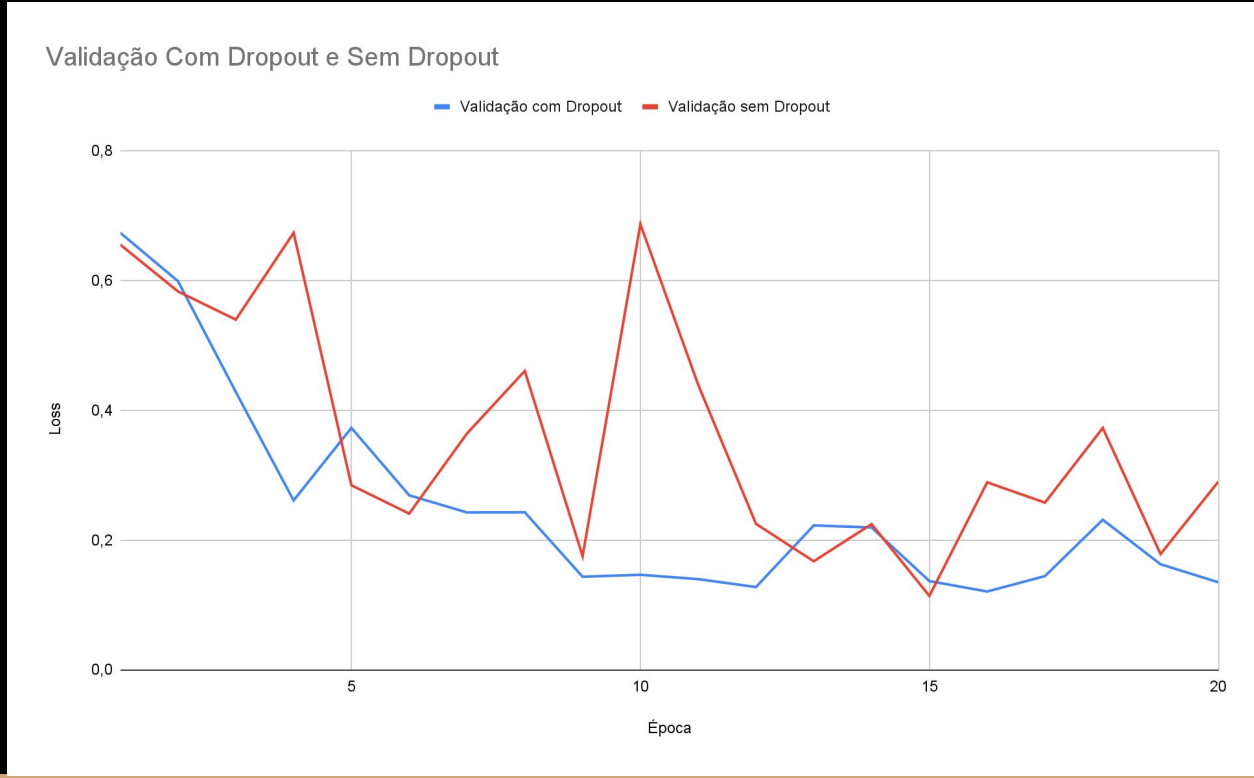
- Compensar a quantidade menor de neurônios ativados durante o treinamento.
- Multiplicar por p a saída dos neurônios durante a predição.

Outra técnica comum é usar Dropout de Monte Carlo.

- Veja mais em Bishop 2024.

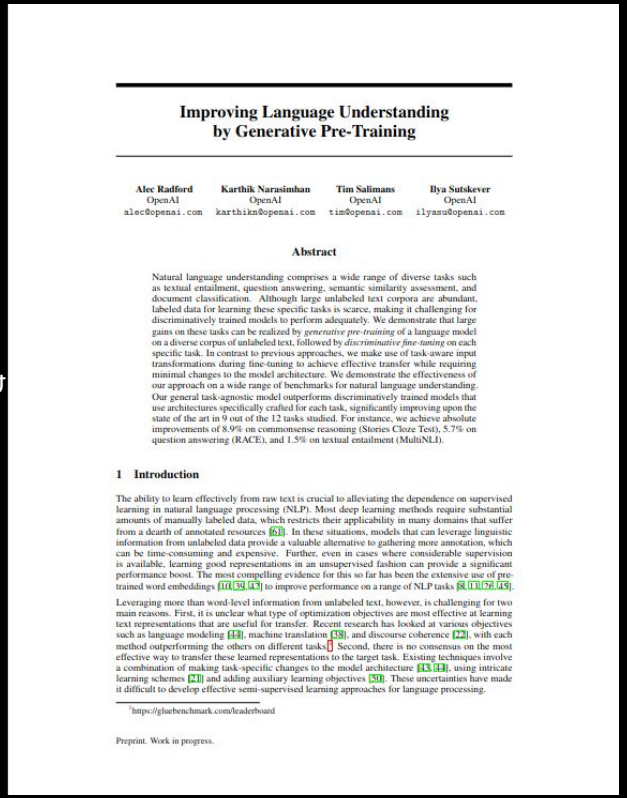
Exemplo

Comparação da validação com e sem dropout do exercício que será dado na aula de hoje.



Exemplo de uso

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. Improving language understanding by generative pre-training. 2018.



Improving Language Understanding by Generative Pre-Training

Alec Radford, OpenAI, alec@openai.com
Karthik Narasimhan, OpenAI, karthikn@openai.com
Tim Salimans, OpenAI, tim@openai.com
Ilya Sutskever, OpenAI, ilyasu@openai.com

Abstract

Natural language understanding comprises a wide range of diverse tasks such as textual entailment, question answering, semantic similarity assessment, and document classification. Although large unlabeled text corpora are abundant, labeled data for learning these specific tasks is scarce, making it challenging for discriminatively trained models to perform adequately. We demonstrate that large gains on these tasks can be realized by *generative pre-training* of a language model on a diverse corpus of unlabeled text, followed by *discriminative fine-tuning* on each specific task. In contrast to previous approaches, we make use of task-aware input transformations during fine-tuning to achieve effective transfer while requiring minimal changes to the model architecture. We demonstrate the effectiveness of our approach on a wide range of benchmarks for natural language understanding. Our general task-agnostic model outperforms discriminatively trained models that use architectures specifically crafted for each task, significantly improving upon the state of the art in 9 out of the 12 tasks studied. For instance, we achieve absolute improvements of 8.9% on commonsense reasoning (Stories Cloze Test), 5.7% on question answering (RACE), and 1.5% on textual entailment (MultiNLI).

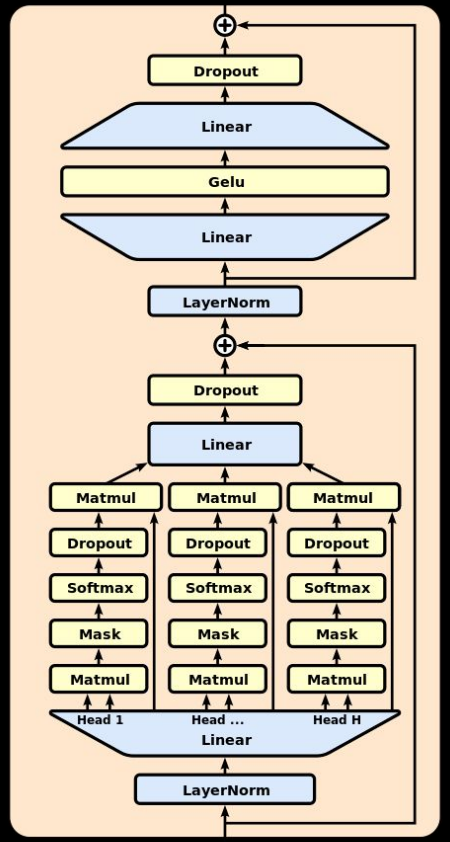
1 Introduction

The ability to learn effectively from raw text is crucial to alleviating the dependence on supervised learning in natural language processing (NLP). Most deep learning methods require substantial amounts of manually labeled data, which restricts their applicability in many domains that suffer from a dearth of annotated resources [61]. In these situations, models that can leverage linguistic information from unlabeled data provide a valuable alternative to gathering more annotation, which can be time-consuming and expensive. Further, even in cases where considerable supervision is available, learning good representations in an unsupervised fashion can provide a significant performance boost. The most compelling evidence for this so far has been the extensive use of pre-trained word embeddings [10, 39, 42] to improve performance on a range of NLP tasks [8, 11, 26, 45].

Leveraging more than word-level information from unlabeled text, however, is challenging for two main reasons. First, it is unclear what type of optimization objectives are most effective at learning text representations that are useful for transfer. Recent research has looked at various objectives such as language modeling [44], machine translation [38], and discourse coherence [22], with each method outperforming the others on different tasks.[1] Second, there is no consensus on the most effective way to transfer these learned representations to the target task. Existing techniques involve a combination of making task-specific changes to the model architecture [43, 44], using intricate learning schemes [21] and adding auxiliary learning objectives [50]. These uncertainties have made it difficult to develop effective semi-supervised learning approaches for language processing.

[1] https://gluebenchmark.com/leaderboard

Preprint. Work in progress.



Arquitetura do GPT original. Original: commons.wikimedia.org/wiki/File:Full_GPT_architecture.svg

Faça você mesmo

Faça o exercício da aula.

Objetivo do exercício:

1. Treinar um MLP.
   a. Já está sendo feito como exemplo. Entenda.
2. Treinar usando Early Stopping.
   a. Já está sendo feito como exemplo. Entenda.
3. Treinar um MLP com Data Augmentation.
4. Treinar um MLP com Dropout.
   a. Dica: como a rede é muito pequena, não exagere no dropout. Use algo como 10% de chance de zerar a máscara do input, e 20% de chance de zerar a máscara das camadas ocultas.
5. Treinar um MLP com Data Augmentation e Dropout.
6. Comparar os resultados.
   a. Dica: em meus testes, sem muitos ajustes, a rede teve 92% de acurácia no dataset de teste.

Esse exercício é uma simplificação do problema da base PKLot. Veja em:

web.inf.ufpr.br/vri/databases/parking-lot-database

Expert Systems with Applications

PKLot – A robust dataset for parking lot classification

Paulo R.L. de Almeida, Luiz S. Oliveira, Alceu S. Britto Jr., Eunelson J. Silva Jr., Alessandro L. Koerich

De Almeida, P. R., Oliveira, L. S., Britto Jr, A. S., Silva Jr, E. J., & Koerich, A. L. PKLot–A robust dataset for parking lot classification. *Expert Systems with Applications*, *42*(11), 4937-4949. 2015.

Referências

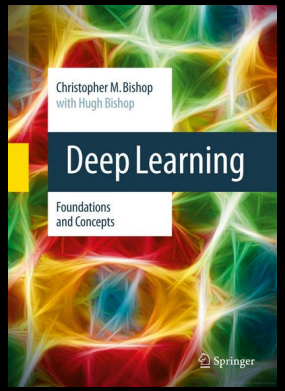
Bishop, C. M., Bishop, H. Deep Learning: Foundations and Concepts. 2023.

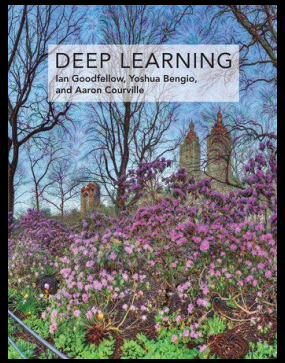
Goodfellow, I., Bengio, Y., Courville, A. Deep Learning. 2016.

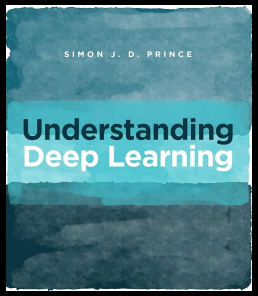
Prince, S. J. Understanding Deep Learning. 2023.

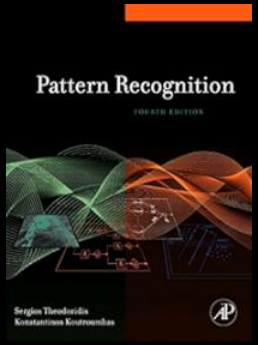
Theodoridis, S., Koutroumbas, K. Pattern Recognition & Matlab Intro. 2010.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*. 2014.









Licença