

Praca Magisterska

Agnieszka Pocha

10 maja 2015

Streszczenie

The goal of this work is to... ..drug design... This is achieved by applying (deep?) convolutional neural networks to the problem.

Spis treści

1	The Problem and the Data	2
1.1	Terminology	2
1.2	Fingerprints	2
1.3	Problem	2
1.4	Datasets	2
1.5	Sparsity	3
1.6	Data representation	3
2	The Model or Deep Convolutional Neural Networks	4
2.1	Deep Neural Networks	4
2.2	Convolutional Neural Networks	4
2.2.1	Motivation	4
2.2.2	Convolution	5
2.2.3	Properties	5
2.2.4	Computation Flow	5
2.2.5	implementationally awesome things	7
2.2.6	Learning Algorithm	7
2.2.7	Extensions	7
2.3	Why was this model chosen	8

Rozdział 1

The Problem and the Data

1.1 Terminology

what is drug design and how does it work? szybko rozwijająca się dziedzina, facing challengeing problems; it's important, new technologies are applied, big data but not complete and from different sources, very expensive and time consuming so modelling is o vita importance

protein, ligand, protein-ligand docking, residues, pharmacophore, interactions, what is the difference between a protein and an amino acid? receptor, donor, active non/inactive protein

1.2 Fingerprints

co to sa fingerprinty i jak się je robi, jakie są problemy, skąd się biorą? po co te fingerprinty, jak te problemy są rozwiązywane, że związki skomplikowane, że na grafach się nie da,

1.3 Problem

drug design, innovative data representation from [2], more details, what exactly am I trying to achieve? *Deep* Convolutional neural networks will be applied to the problem.

1.4 Datasets

The data consists of multiple datasets, each describing reactions between a single protein and multiple ligands. Each dataset consists of four dimensions described by: the number of ligands, length of the protein, 6 standard pharmacophore features of ligand and 9 types of interactions with amino acid[2]. One data sample can be seen as a 3-dimensional matrix describing how a single ligand bounds with a specific protein. The 3 dimensions are: the length of the protein (number of its residues), 6 standard pharmacophore features and 9 types of interactions

with amino acid.

LUB

The data consists of multiple datasets, each describing reactions between a single protein and multiple ligands and providing information whether (the *combination* is active or not). Each dataset consists of four dimensions described by: the number of ligands, length of the protein, 6 standard pharmacophore features of ligand and 9 types of interactions with amino acid[2]. One data sample can be seen as a 3-dimensional matrix describing how a single ligand bounds with a specific protein. The 3 dimensions are: the length of the protein (number of its residues), 6 standard pharmacophore features and 9 types of interactions with amino acid. The labels are...

{obrazek}

The 6 pharmacophore features are: hydrogen bond acceptor, hydrogen bond donor, hydrophobic, negatively charged group, positively charged group, aromatic. The 9 types of interactions with amino acid are: any, with a backbone, interaction with sidechain, polar, hydrophobic, hydrogen bond acceptor, hydrogen bond donor, charged interaction, aromatic.

The values constituting the ... dataset are discrete, namely: 0, 1 and 2. 0 means there is no interaction of specific kind. The labels are represented as 0 (not active) or 1 (active).

1.5 Sparsity

check if the data is sparse, if yes then state that it is and explain why

1.6 Data representation

Each data sample is represented as a vector of $r * 6 * 9$ length, where r is the length of the protein. Data samples *są zebrane w* a dataset. Each dataset describes how a certain protein reacts with amino acid;.

why was this particular fingerprint representation chosen

Rozdział 2

The Model of Deep Convolutional Neural Networks

2.1 Deep Neural Networks

DNN

2.2 Convolutional Neural Networks

The simplest definition of convolutional neural networks is probably: neural networks that take advantage of using the convolution operation. Usually the CNN is *conceptually* divided into two subnetworks: first subnetwork is *built from* convolutional layers and is responsible for feature extraction, the second one is a classical neural network, e.g. multilayer perceptron. Its aim is to *poprawnie* classify the examples taking as input the features extracted by the previous subnetwork.

{obrazek}

In this section I will give motivation that stands behind using such networks, *explain/define* what is the convolution operation, and give a detailed explanation of CNNs and its properties. Finally, I will describe what problems might arise while learning a CNN model and how to prevent them. The learning algorithm for CNNs will be given.

2.2.1 Motivation

Convolutional neural networks are *mostly* applied to image recognition, video analysis and natural language processing problems. This attempts *are often successful/often give better results than (any other) models*.

2.2.2 Convolution

Convolution operation takes as *operands* two functions (dziedzina? zbiór wartości?) and return a new function (dziedzina? zbiór wartości?) as a result. Mathematically, convolution is defined as:

$$c(t) = \int_{-\infty}^{\infty} f(x)g(t-x)dx$$

In the equation above we can see c - a function returned by convolution operation - that is evaluated in point t . c is defined as an integral over two other functions. f is often called an input, while g is often referred to as a kernel.

It is often useful to see kernels as feature extractors. why kernels can be seen as feature extractors?

2.2.3 Properties

spatial invariance, napisac jak zmieniaja sie wymiary macierzy podczas konwolucji, pooling, itd. pooling shape, pooling stride,

2.2.4 Computation Flow

As stated above, CNNs can be conceptually divided into two subnetworks. In this subsection I will describe how the signal is processed within the convolutional subnetwork. I will not *dig into* the classifying subnetwork as any neural network that can be used to classify objects might be used. Many such networks exist*s* and they are well described in the literature.

In each layer there are three *podstawowe* operations *wykonywane*. Firstly, the input is convoluted with a kernel matrix. The result of this operation is an input to the activation function. If the input is a matrix *(and it usually/always is a matrix)* each element forms a single input to the activation function *(a może da się inaczej?)*. Finally, the pooling is applied to the result.

{obrazek} może też wzorek? pooling(sigmoid(convolution(input)))

One might also think imagine three consecutive separate layers: a convolutional layer, a *widely-used/classical* layer that applies activation function *(activation layer)* and finally, a pooling layer.

In the following subsections I will describe in details how each of this operation exactly works. *Szczególna uwaga zostanie zwrócona na to, jak zmieniają się wymiary danych na każdym etapie*

{obrazek} chyba ten u góry tylko

Convolution for neural networks

naming: input, kernel, output, powiedziec o tym, co sie dzieje na brzegach (kernel nie lazi), i jak sobie z tym mozna radzic (walce, torusy, dopisywanie zerami

(zero-pad), lustrzane odbicie, godzenie się na mniejszą macierz. Valid, same and full convolution (MATLAB terminology) Opisać jakie rozwiązanie zostało użyte przez nas. one kernel = one feature, therefore typically many kernels are used. Also: kernels in consecutive layers work on the outputs of the previous layers → in each layer more complicated, i.e. constructed from simpler ones, features are used

Activation function

różne możliwe typy

Pooling

Pooling is an operation that takes as input multiple values and returns *the statistic(s) of these values*. It is usually applied on a matrix taking as input the submatrices and returning a single value as a result. 3 most common *(citation needed?)* types of pooling are:

- max pooling - the max value is returned
- average pooling - the average value is returned
- weighted average pooling - the weighted average is returned. Weights are usually *(citation needed?)* defined by the distance from *what?*

types: Bengio, 181, pierwszy pełny akapit, pooling shape and stride → boost computational efficiency.

{obrazek} jak wygląda max pooling

Pooling is defined not only by its type but also by its size and stride. The size of pooling defines how many values will be taken as an input - the bigger the size of pooling, the more information is *lost*. The pooling stride defines how the submatrices are *rozmieszczone względem siebie*. Fig *???* shows how *does it work*.

{obrazek} pooling size and stride

how does pooling give us some invariance?

{obrazek} dlaczego daje nam invariance

is pooling subsampling and if yes then why is pooling subsampling?

Summary

input goes through three stages, downchanging its size

2.2.5 implementationally awesome things

sparse interactions

because kernel is smaller than data so it not kazdy z kazdym but some with some (sparse) \rightarrow computational boost, kernel is small and moved around input - less parameters, instead of a big matrix we store a small one that runs over the data

{obrazek}

parameter sharing

Connected to the fact that we move the convolution kernel around

{obrazek} a może nie?

equivariant representations

equivariance - property of *what?* meaning that if the input changes that output changes the same way. $f(g(x)) = g(f(x))$ Intuition about it: detecting feature in a particular place - feature elsewhere - we find it elsewhere. To what types of transformation is convolution equivariant and to which transformations it isn't?

2.2.6 Learning Algorithm

A *zmieniona wersja* of backpropagation algorithm has been provided to *include the changes that must be applied because of* the convolution operation and avoid the diminishing gradient flow. In this subsection the *zmieniona wersja* of backpropagation is given and the problem of diminishing gradient flow is *addressed*.

The problems with a classical backpropagation

diminishing gradient flow, niedouczenie się, przeuczenie się, obczaić co o tym mówił Larochelle, on to chyba jednak mówił o głębokich. Wtedy to i tak napisać i przerzucić do głębokich.

Diminishing gradient flow

co to jest, skąd się bierze, można się wesprzeć wykładami Larochelle, on poleca dużo papierów zawsze.

Backpropagation

See (Goodfellow, 2010) from Bengio

2.2.7 Extensions

dropout/dropconnect method, activation functions for dropout, other things from the Dutch paper

2.3 Why was this model chosen

... Having said that kernels might be used as feature extractors it's worth considering what kinds of features might be discovered in the provided data. ...

Bibliografia

- [1] Yoshua Bengio, Ian J. Goodfellow, Aaron Courville - *Deep Learning*
- [2] Stefan Mordalski, Igor Podolak, Andrzej J. Bojarski - *2D SIFT - a matrix of ligand-receptor interactions*