



Machine Learning for Theorem Proving

Lecture 8 (VU)

Cezary Kaliszyk

Overview

Last Lecture

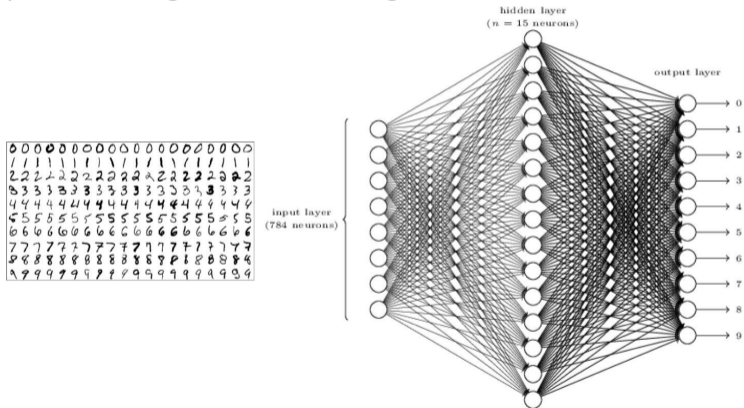
- Syntactic methods for premise selection
- Random forests

Today

- short reminder on neural networks
- deep learning for premise selection
- type inference

Neural Network: Recognize a handwritten character

We create an input layer that takes as input the pixels of an image. Intermediate layers use weights to compute intermediate values. The values in the output layer are then compared to predict the digit. Measure: recognition rate.



Such networks work very well i.e. on the MNIST dataset.

Neural Networks: Third edition

Modelling of Neurophysiological Networks (1950s – 1960s)

- Simple networks of individual perceptrons, with basic learning
- Severe limitations

[Minsky,Papert]

Paralled Distributed Processing (1990s)

- rejuvenated interest
- But statistical algorithms were comparably powerful (SVM)

[Rumelhart,MacClelland]

Deep Learning (2010s)

- Data-oriented algorithms
- Data and processing were a limitation before

Expressiveness of multilayer perceptron networks

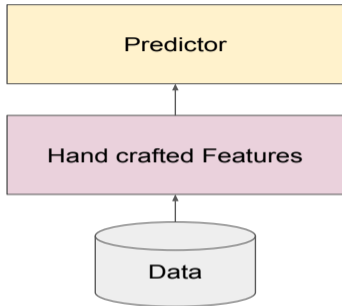
Perceptrons implement linear separators, but:

- Every continuous function modeled with three layers (this means just one hidden)
- Every function can be modeled with four layers
- But the layers are assumed to be arbitrarily large (and indeed they would be enormous)!

These results have been recently formalized

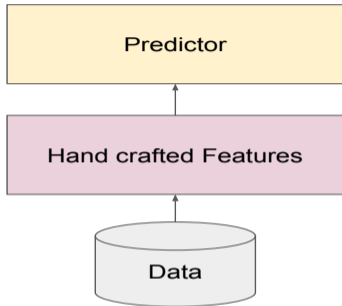
[*Bentkamp'17*]

Deep Learning vs Shallow Learning

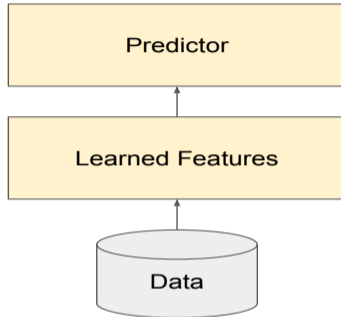


Traditional machine learning

Deep Learning vs Shallow Learning

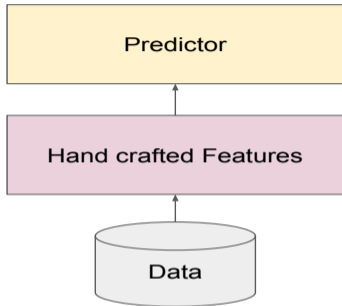


Traditional machine learning



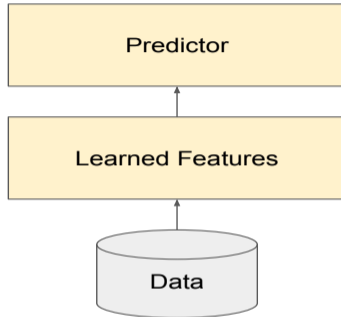
Deep Learning

Deep Learning vs Shallow Learning



Traditional machine learning

- Mostly convex, provably tractable
- Special purpose solvers
- Non-layered architectures

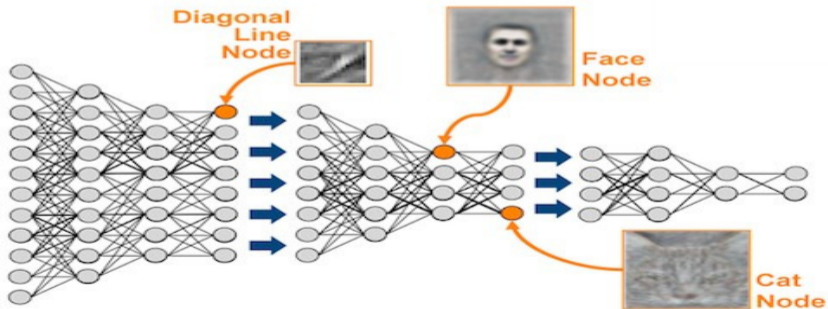


Deep Learning

- Mostly NP-Hard
- General purpose solvers
- Hierarchical models

Deep Learning

- Lots of successes with training “deep” networks (multiple layers)
- Image classification: sequential extraction of information
- Big data, fast processing, some tricks
 - ReLU, Faster Propagation, Autoencoding, Dropout, Residual connections, ...



Recurrent Neural Networks

An alternatives to convolutions are recurrent neural networks, where the input is processed sequentially and the results of processing of a part of the input is used as part of the input of the processing of a next one.

Recurrent Neural Networks (RNN)

process sequences by feeding back the output into the next input

Long-Short Term Memory (LSTM)

add forgetting to RNNs

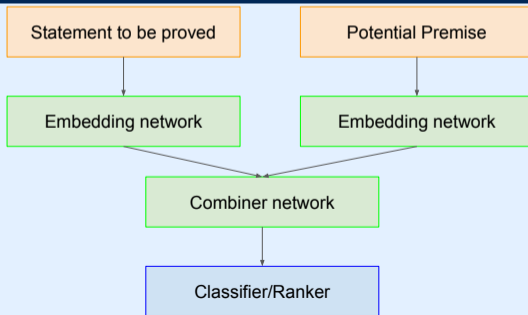
[Hochreiter]

These can be used to build end-to-end learning, for example sequence-to-sequence models used in language translation today.

Simple classifier on top of concatenated embeddings

- different model of premise selection
- trained to estimate usefulness
- positive and negative examples

Architecture



Deep Learning for Mizar Lemma Selection

- No hand-engineered features
- Comparison of various neural architectures
- Semantic-aware definition embeddings
- Complementary to previous approaches
- Can be ensembled

DeepMath: Dataset

The Mizar Mathematical Library (MML) is a corpus of formal mathematical proofs, containing 57,917 theorems from a wide variety of mathematical subjects. We worked with a version converted to first order logic in the TPTP format.

```
:: t99_jordan: Jordan curve theorem in Mizar
for C being Simple_closed_curve holds C is Jordan;

:: Translation to first order logic
fof(t99_jordan, axiom, (! [A] : ( (v1_topreal2(A) & m1_subset_1(A,
k1_zfmisc_1(u1_struct_0(k15_euclid(2)))) => v1_jordan1(A)) ) ) ).
```

Figure 1: (top) The final statement of the Mizar formalization of the Jordan curve theorem. (bottom) The translation to first-order logic, using name mangling to ensure uniqueness across the entire corpus.

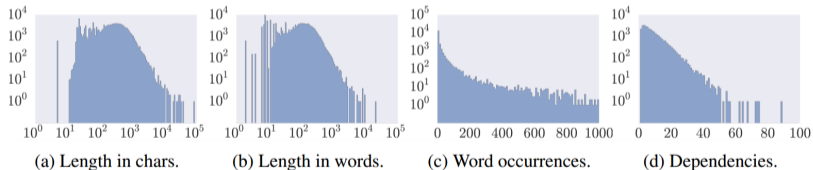


Figure 2: Histograms of statement lengths, occurrences of each word, and statement dependencies in the Mizar corpus translated to first order logic. The wide length distribution poses difficulties for RNN models and batching, and many rarely occurring words make it important to take definitions of words into account.

DeepMath: Problem, Metric, Model

Definition (Premise selection problem). *Given a large set of premises \mathcal{P} , an ATP system A with given resource limits, and a new conjecture C , predict those premises from \mathcal{P} that will most likely lead to an automatically constructed proof of C by A .*

$$\text{aMRR} = \text{mean}_C \max_{P \in \mathcal{P}_{\text{test}}(C)} \frac{\text{rank}(P, \mathcal{P}_{\text{avail}}(C))}{|\mathcal{P}_{\text{avail}}(C)|}$$

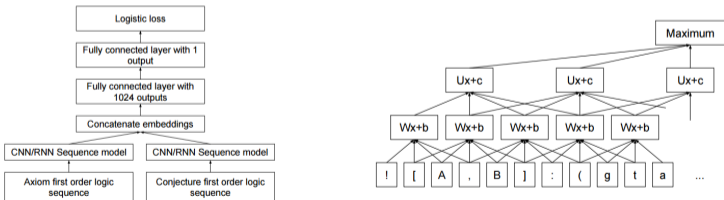


Figure 3: (left) Our network structure. The input sequences are either character-level (section 5.1) or word-level (section 5.2). We use separate models to embed conjecture and axiom, and a logistic layer to predict whether the axiom is useful for proving the conjecture. (right) A convolutional model.

DeepMath: Architectures

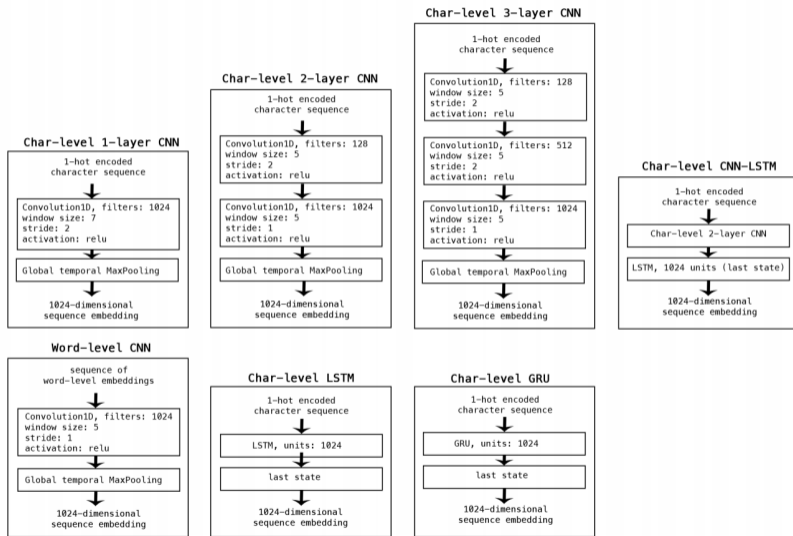


Figure 4: Specification of the different embedder networks.

DeepMath: Results

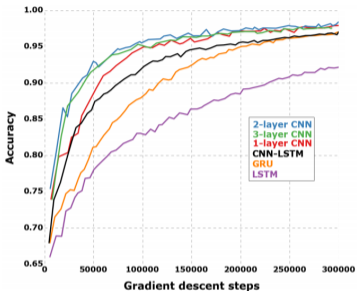
Cutoff	<i>k</i> -NN Baseline (%)	char-CNN (%)	word-CNN (%)	def-CNN-LSTM (%)	def-CNN (%)	def+char-CNN (%)
16	674 (24.6)	687 (25.1)	709 (25.9)	644 (23.5)	734 (26.8)	835 (30.5)
32	1081 (39.4)	1028 (37.5)	1063 (38.8)	924 (33.7)	1093 (39.9)	1218 (44.4)
64	1399 (51)	1295 (47.2)	1355 (49.4)	1196 (43.6)	1381 (50.4)	1470 (53.6)
128	1612 (58.8)	1534 (55.9)	1552 (56.6)	1401 (51.1)	1617 (59)	1695 (61.8)
256	1709 (62.3)	1656 (60.4)	1635 (59.6)	1519 (55.4)	1708 (62.3)	1780 (64.9)
512	1762 (64.3)	1711 (62.4)	1712 (62.4)	1593 (58.1)	1780 (64.9)	1830 (66.7)
1024	1786 (65.1)	1762 (64.3)	1755 (64)	1647 (60.1)	1822 (66.4)	1862 (67.9)

Table 1: Results of ATP premise selection experiments with hard negative mining on a test set of 2,742 theorems.

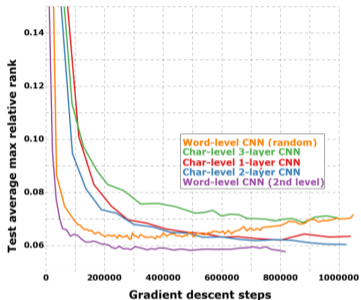
- E-prover proved theorem percentages
- Union of all methods: 80.9%
- Union of deep network methods: 78.4%

Note: These results are good, but the improvement over a simple *k*-NN is just a few percent. At the same time preparing *k*-NN on Mizar data takes 2 seconds on a single CPU, while training the [Alemi+] neural network took one week on multiple GPUs

DeepMath: Accuracy

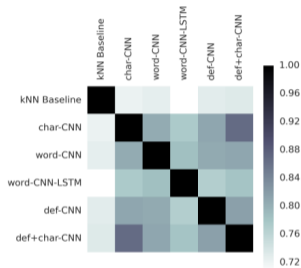


(a) Training accuracy for different character-level models without hard negative mining. Recurrent models seem underperform, while pure convolutional models yield the best results. For each architecture, we trained three models with different random initialization seeds. Only the best runs are shown on this graph; we did not see much variance between runs on the same architecture.



(b) Test average max relative rank for different models without hard negative mining. The best is a word-level CNN using definition embeddings from a character-level 2-layer CNN. An identical word-embedding model with random starting embedding overfits after only 250,000 iterations and underperforms the best character-level model.

DeepMath: Statistics



(a) Jaccard similarities between proved sets of conjectures across models. Each of the neural network model prediction are more like each other than those of the k -NN baseline.

Model	Test min average relative rank
char-CNN	0.0585
word-CNN	0.06
def-CNN-LSTM	0.0605
def-CNN	0.0575

(b) Best sustained test results obtained by the above models. Lower values are better. This was monitored continuously during training on a holdout set with 400 theorems, using all true positive premises and 128 randomly selected negatives. In this setup, the lowest attainable max average relative rank with perfect predictions is 0.051.

DeepMath: Hard Negatives

Initially as positive examples we use true dependencies, as negatives we use random things that are not among our known dependencies. This still produced results worse than kNN. A major necessary infrastructure required was *Hard Negatives*.

This means learn the negative examples that are not predicted correctly. Rather than have 2 queues: half positives and half negatives, we have three queues: Positives, negatives, and hard negatives, that is negatives that we have not predicted correctly.

Other neural network architectures

Other neural network architectures

Examples

- Seq2Seq
- GNN

How could we use these for theorem proving tasks?

Next problem: Lemma Usefulness

The next problem that we will look at will be intermediate lemma usefulness

In large proofs processed by an ITP (actually also by an ATP) there are many intermediate steps. Some of these are particularly useful (can they be reused), and some are useless (will not be necessary in the final proof as they are tautologies etc).

Can we use learning methods to predict the particularly useful ones (to name them and keep them for reuse) and can we predict the useless ones (to avoid work)?

We first look at a deep learning approach to the problem, and later at other (more proving) specific learning methods.

Learning Lemma Usefulness

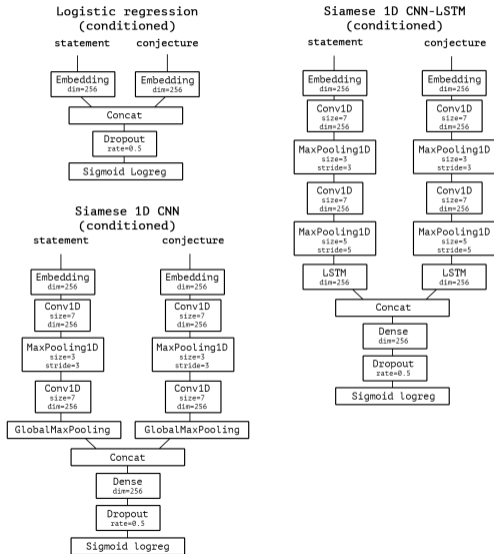
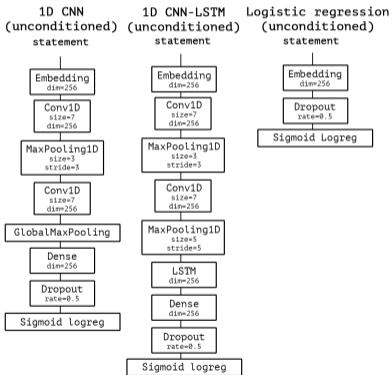
HOLStep Dataset

- Intermediate steps of the Kepler proof
- Only relevant proofs of reasonable size
- Annotate steps as useful and unused
 - Same number of positive and negative
- Tokenization and normalization of statements

Statistics

	Train	Test	Positive	Negative
Examples	2013046	196030	1104538	1104538
Avg. length	503.18	440.20	535.52	459.66
Avg. tokens	87.01	80.62	95.48	77.40
Conjectures	9999	1411	-	-
Avg. deps	29.58	22.82	-	-

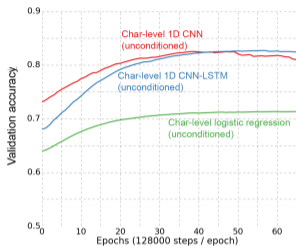
Considered Models



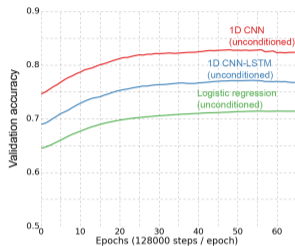
Baselines (Training Profiles)

unconditioned

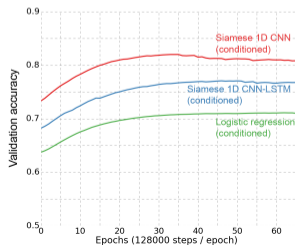
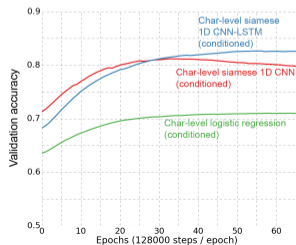
char-level



token-level



cojecture
conditioned



Summary

This Lecture



- deep learning for theorem proving

Next

- evaluation metrics
- start on ATPs

Additional Literature (not required)

The formalization of deep learning and deep learning for premise selection.

-  Alexander Bentkamp, Jasmin Christian Blanchette, and Dietrich Klakow.
A formal proof of the expressiveness of deep learning.
In Mauricio Ayala-Rincón and César A. Muñoz, editors, *Interactive Theorem Proving - 8th International Conference*, volume 10499 of *LNCS*, pages 46–64. Springer, 2017.
-  Geoffrey Irving, Christian Szegedy, Alexander A. Alemi, Niklas Eén, François Chollet, and Josef Urban.
Deepmath - deep sequence models for premise selection.
In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2235–2243, 2016.

Work Here / Homework

Direct Neural Network Use

- Would it be possibly to apply a modern neural network to premise selection directly?
- What issues do you see?

Lemma Selection

- Run an ATP on a larger problem to produce a number of intermediate statements
- Are they useful for the conjecture using your algorithm implemented in one of the previous homeworks?
(For features you can split the string on spaces for example)