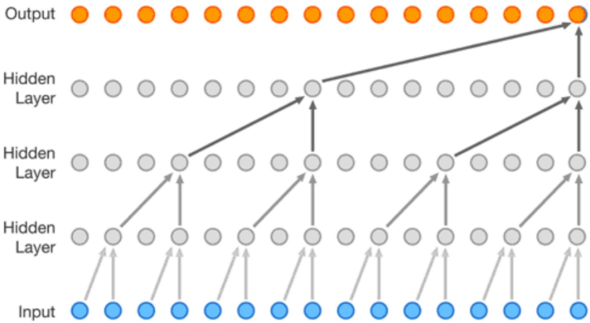
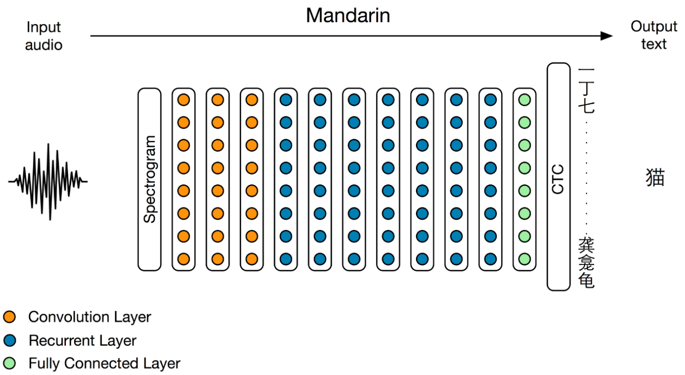
Wide learning in language modeling

Harald Baayen
Quantitative Linguistics Group
University of Tübingen

wavenet



deep speech


Mandarin
Input audio
Output text
Spectrogram
CTC
一
丁
七
龚
龛
龟
猫
Convolution Layer
Recurrent Layer
Fully Connected Layer

linguists have a bad reputation

Every time I fire a linguist,
the performance of the speech recognizer goes up.

(F. Jelinek)

research strategies for linguistics

1. dismiss the results from engineering as irrelevant

   or

2. take the units of deep convolution networks
   to be the true representations of sound units (phonemes) and
   minimal meaning-bearing units (morphemes)
   (the heritage of the logic tradition in linguistics)

   or

3. set aside the traditional “hidden” constructs of linguistics
   and take a fresh look at how language might work

overview

1. word learning in baboons
2. auditory word recognition
3. inflecting Latin verbs

modeling tool

- discrimination learning with simple two-layer networks
- incremental multivariate multiple regression

the bandwagon of deep learning

Grainger et al. (Science, 2012)

French baboons doing lexical decision in English learn up to 300 words

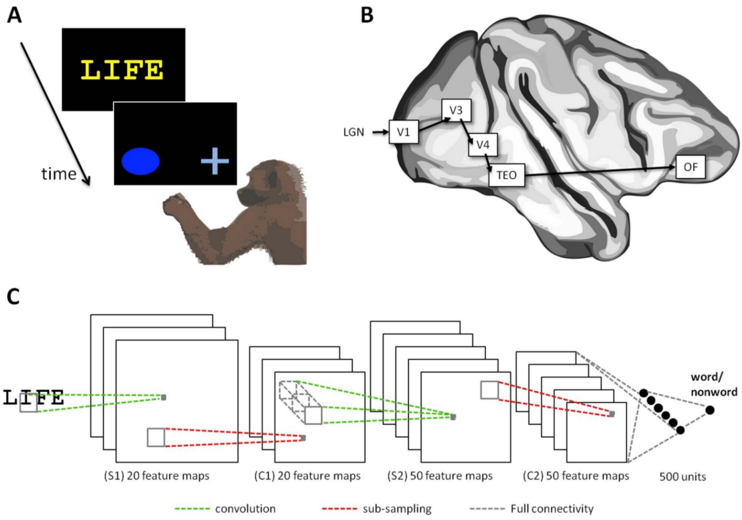
TORE EFTD WEND ULKH BANG BANG ULNX TORE PSHA AHMF
BOOR KRBA KRBO WEND BANG KRBU IMMF BANG PSMI OHMF

TORE

EFTD

WEND

a deep convolution network
letters and letter pairs ‘emerge’ on hidden layers



problems

1. pigeons can also do the task — with very different brains

2. the data was never properly analysed,
yet strong claims were put forward, for instance,
baboons are supposed to learn letter pairs and triplets,
but not words

a GAMM (mgcv) for baboon learning over time

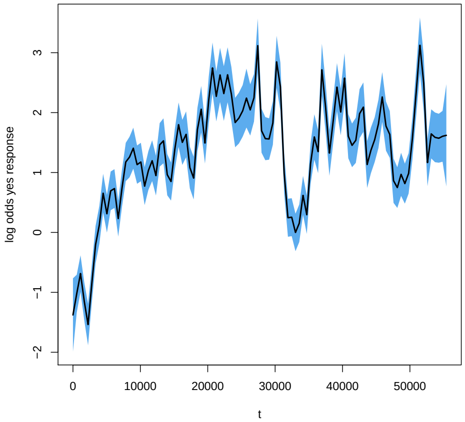
$y_t \underset{\text{ind}}{\sim} \text{binom}(\exp(\eta_t)/\{1+\exp(\eta_t)\}, 1)$ where $\eta_t = \beta_0 + s(t) + z_{w(t)}$

$z_{w(t)} \sim N(0, \sigma^2)$

```
Response ~ s(t, k = 200) + s(word, bs = "re")

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.3541     0.1004   13.48   <2e-16

Approximate significance of smooth terms:
           edf Ref.df Chi.sq p-value
s(t)    105.05  128.6   1813  <2e-16
s(word)  77.39   86.0   1646  <2e-16
```


log odds yes response
−2
−1
0
1
2
3
0
10000
20000
30000
40000
50000
t

modeling with ‘wide’ learning

$$\mathbf{WC} = \mathbf{T}$$

$$\mathbf{W}^t = \mathbf{W}^{t-1} + \boldsymbol{\Delta}_{rw}$$

$$\Delta_{ij} = \begin{cases} 0 & \text{if } \textsc{absent}(c_i, t) \\ \alpha\left(1 - \sum_{\textsc{present}(c_k,\, t)} w_j\right) & \text{if } \textsc{present}(c_i, t)\ \&\ \textsc{present}(o_j, t) \\ \alpha\left(0 - \sum_{\textsc{present}(c_k,\, t)} w_j\right) & \text{if } \textsc{present}(c_i, t)\ \&\ \textsc{absent}(o_j, t) \end{cases}$$

$$(\alpha \ll 1)$$

Rescorla-Wagner learning rule (ndl; ndl2; pyndl)

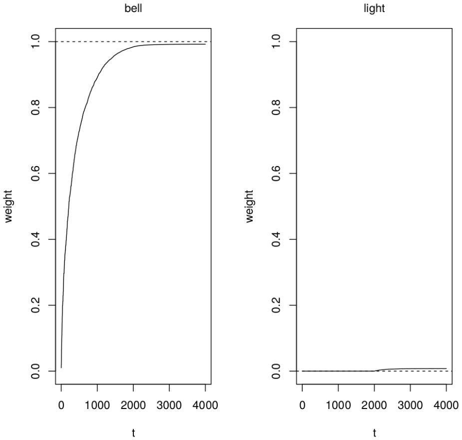
blocking in learning

- Pavlov's dog: trained to expects food when a bell rings
- continue training with flashing a light while ringing the bell
- then: flash the light, but don't ring the bell
- will the dog drool?

blocking in learning

- Pavlov's dog: trained to expects food when a bell rings
- continue training with flashing a light while ringing the bell
- then: flash the light, but don't ring the bell
- will the dog drool?
- no

blocking in learning



$$\Delta_{ij} = \begin{cases} 0 & \text{if } \textsc{absent}(c_i, t) \\ \alpha\left(1 - \sum_{\textsc{present}(c_k,\, t)} w_j\right) & \text{if } \textsc{present}(c_i, t) \;\&\; \textsc{present}(o_j, t) \\ \alpha\left(0 - \sum_{\textsc{present}(c_k,\, t)} w_j\right) & \text{if } \textsc{present}(c_i, t) \;\&\; \textsc{absent}(o_j, t) \end{cases}$$

Rescorla-Wagner and Widrow-Hoff

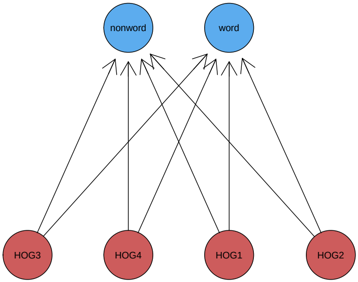
incremental regression using the Widrow-Hoff learning rule

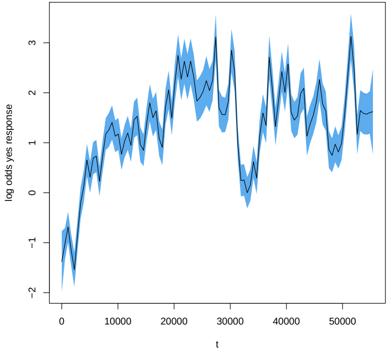
$$\boldsymbol{\Delta}_{wh} = \eta\{\mathbf{a}(\mathbf{o} - \mathbf{a}^T\mathbf{W})\}.$$

Bernard Widrow & Marcian. E. Hoff (1960). Adaptive switching circuits, 1960 WESCON Convention Record Part IV, p. 96–104.

application to baboon word learning

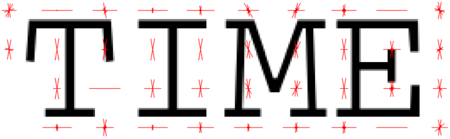
TORE EFTD WEND ULKH BANG BANG ULNX TORE PSHA AHMF
BOOR KRBA KRBO WEND BANG KRBU IMMF BANG PSMI OHMF





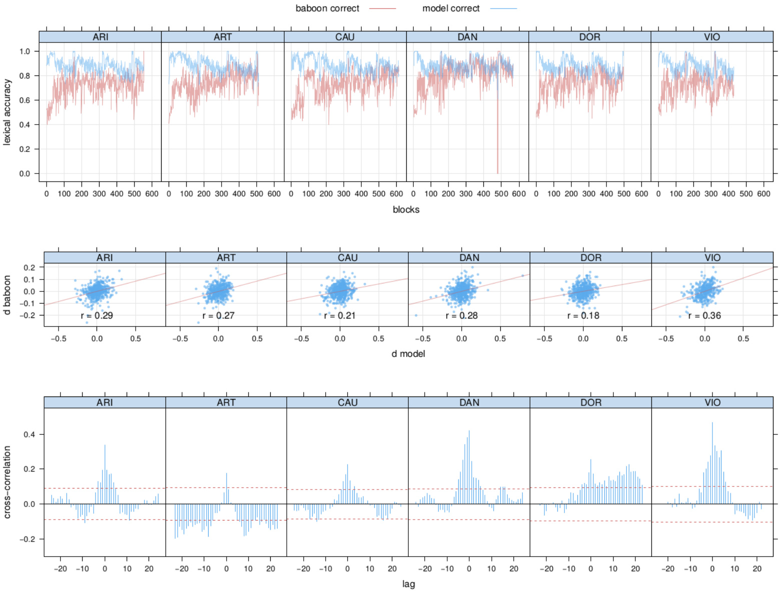
15,149 HOG cues $\times$ 2 outcomes

histograms of oriented gradients (HOG) features



Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), volume 1, pages 886–893 vol. 1.

OpenImageR: HOG()


baboon correct
model correct
ARI
ART
CAU
DAN
DOR
VIO
lexical accuracy
blocks
d baboon
d model
r = 0.29
r = 0.27
r = 0.21
r = 0.28
r = 0.18
r = 0.36
cross-correlation
lag

a less trivial task: auditory word recognition

Johnson (2004): Counts for English indicate that one out of 20 words is spoken with at least one syllable from the canonical form missing, and that up to 20% of the content words and up to 40% of the function words have at least one phone missing.

a less trivial task: auditory word recognition

Johnson (2004): Counts for English indicate that one out of 20 words is spoken with at least one syllable from the canonical form missing, and that up to 20% of the content words and up to 40% of the function words have at least one phone missing.

- hleres

a less trivial task: auditory word recognition

Johnson (2004): Counts for English indicate that one out of 20 words is spoken with at least one syllable from the canonical form missing, and that up to 20% of the content words and up to 40% of the function words have at least one phone missing.

- hleres hilarious

a less trivial task: auditory word recognition

Johnson (2004): Counts for English indicate that one out of 20 words is spoken with at least one syllable from the canonical form missing, and that up to 20% of the content words and up to 40% of the function words have at least one phone missing.

- hleres hilarious
- yeshay

a less trivial task: auditory word recognition

Johnson (2004): Counts for English indicate that one out of 20 words is spoken with at least one syllable from the canonical form missing, and that up to 20% of the content words and up to 40% of the function words have at least one phone missing.

- hleres hilarious
- yeshay yesterdag

a less trivial task: auditory word recognition

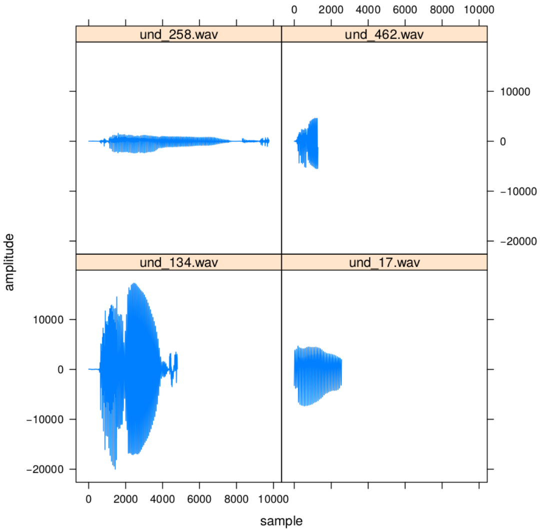
Johnson (2004): Counts for English indicate that one out of 20 words is spoken with at least one syllable from the canonical form missing, and that up to 20% of the content words and up to 40% of the function words have at least one phone missing.

- hleres hilarious
- yeshay yesterdag
- wün

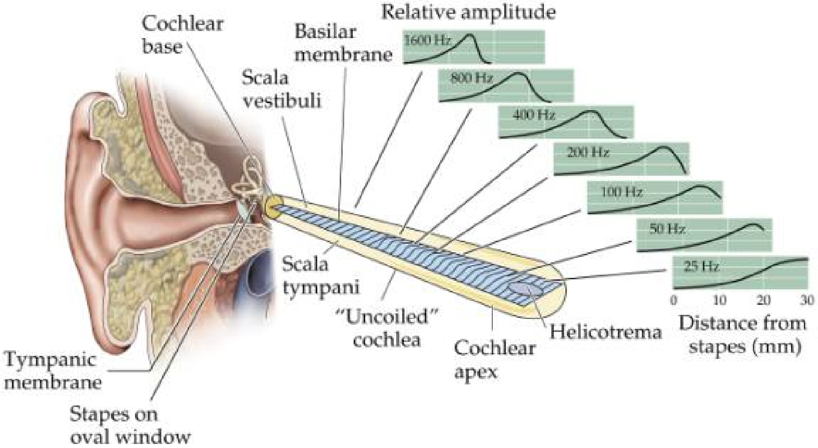
a less trivial task: auditory word recognition

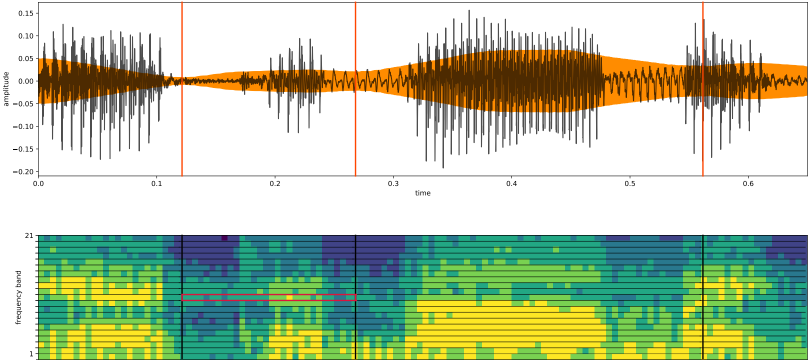
Johnson (2004): Counts for English indicate that one out of 20 words is spoken with at least one syllable from the canonical form missing, and that up to 20% of the content words and up to 40% of the function words have at least one phone missing.

- hleres hilarious
- yeshay yesterdag
- wün würden



upper left: ʊnt^h, upper right: ʊn,
lowel left: ʊnt^h, lower right: n.


Relative amplitude
Cochlear base
Basilar membrane
Scala vestibuli
1600 Hz
800 Hz
400 Hz
200 Hz
100 Hz
50 Hz
25 Hz
0
10
20
30
Distance from stapes (mm)
Scala tympani
"Uncoiled" cochlea
Cochlear apex
Helicotrema
Tympanic membrane
Stapes on oval window



amplitude
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
0.0
0.1
0.2
0.3
0.4
0.5
0.6
time
frequency band
21
1

frequency band summary (FBS) features

band1start1median2min1max4end2part1

```
lowest frequency band
first chunck

first intensity value = 1
median = 2
minimum = 1
maximum = 4
final value = 2
```

89,333 different FBS for the 20 hours of speech (246,625 word tokens) in the GECO corpus

Arnold D. `AcousticNDLCoder`: Coding sound files for use with NDL; 2016. R package version 0.1.1.

evaluation on random sample of 1000 (often reduced) words

- **human performance**
  - 20% – 40%
- **model performance**
  - 20–25%

Inflecting Latin verbs

	SINGULAR			PLURAL		
	1	2	3	1	2	3
PRESENT	vocoo	vocaas	vocat	vocaamus	vocaatis	vocant
PAST	vocaabam	vocaabaas	vocaabat	vocaabaamus	vocaabaatis	vocaabant
⋮	⋮	⋮	⋮	⋮	⋮	⋮
PLUPERFECT	vocaavissem	vocaavissees	vocaavisset	vocaavisseemus	vocaavisseetis	vocaavissent

Latin inflectional classes

CLASS I	CLASS II	CLASS III	CLASS IV	tense	voice	mood
vocoo	terreoo	carpoo	audioo	present	active	ind
vocem	terream	carpiam	audiam	present	active	subj
vocor	terreor	carpor	audior	present	passive	ind
vocer	terrear	carpiar	audiar	present	passive	subj
vocaaboo	terreeboo	carpam	audiam	future	active	ind
vocaabor	terreebor	carpar	audiar	future	passive	ind
vocaabam	terreebam	carpeebam	audieebam	past	active	ind
vocaarem	terreerem	carperem	audiirem	past	active	subj
vocaabar	terreebar	carpeebar	audieebar	past	passive	ind
vocaarer	terreerer	carperer	audiirer	past	passive	subj
vocaavii	terruii	carpsii	audiivii	perfect	active	ind
vocaaverim	terruerim	carpserim	audiiverim	perfect	active	subj
vocaaveram	terrueram	carpseram	audiiveram	pluperfect	active	ind
vocaavissem	terruissem	carpsissem	audiivissem	pluperfect	active	subj

Latin verb conjugations

class I
: theme vowel a
: future with exponent b
: perfect with exponent v

class II
: theme vowel e
: future with exponent b
: perfect with u

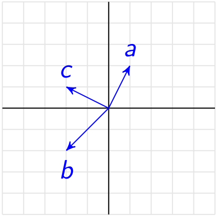
class III
: no theme vowel
: future with exponent am
: irregular past participle and perfect

class IV
: theme vowel i
: future with exponent am
: perfect with exponent v

core idea

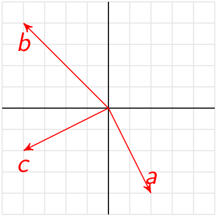
1. represent words' forms as numeric vectors
2. represent words' meanings as numeric vectors
3. use the mathematics of linear transformations to move between form and meaning

$$\boldsymbol{C} = \begin{pmatrix} 1 & 2 \\ -2 & -2 \\ -2 & 1 \end{pmatrix} \qquad\qquad \boldsymbol{S} = \begin{pmatrix} 2 & -4 \\ -4 & 4 \\ -4 & -2 \end{pmatrix}$$



$$\xleftarrow[\boldsymbol{G}]{\begin{pmatrix} 0.5 & 0 \\ 0 & -0.5 \end{pmatrix}}$$

$$\xrightarrow[\begin{pmatrix} 2 & 0 \\ 0 & -2 \end{pmatrix}]{\boldsymbol{F}}$$



words' forms: the row vectors of $\boldsymbol{C}$
words' meanings: the row vectors of $\boldsymbol{S}$

numeric vectors for words' forms

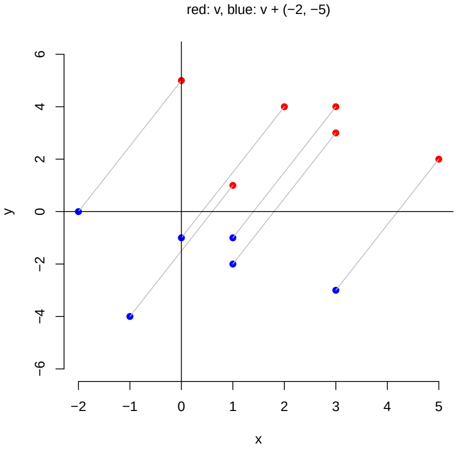
```
##            #vo voc oco coo oo# oca caa aas as# cat at# aam
## vocoo        1   1   1   1   1   0   0   0   0   0   0   0
## vocaas       1   1   0   0   0   1   1   1   1   0   0   0
## vocat        1   1   0   0   0   1   0   0   0   1   1   0
## vocaamus     1   1   0   0   0   1   1   0   0   0   0   1
## vocaatis     1   1   0   0   0   1   1   0   0   0   0   0
## vocant       1   1   0   0   0   1   0   0   0   0   0   0
## clamoo       0   0   0   0   1   0   0   0   0   0   0   0
## clamaas      0   0   0   0   0   0   0   1   1   0   0   0
```

binary vectors specifying which letter triplets (or triphones) are present in a word (1) and which are absent (0)

numeric vectors for words’ meanings

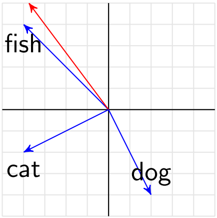
```
##               S1    S2     S3    S4    S5    S6    S7    S8
## vocoo       1.59  4.92 -17.84 24.26 -7.73  8.47  0.35 10.07
## vocaas      6.35  4.88 -11.26 29.83 -8.82  7.08 -4.92  9.44
## vocat      -1.85  3.01  -8.26 26.58 -4.95  3.79  2.97  4.85
## vocaamus    3.31 10.41 -25.43 20.01 -6.08  1.16 -3.33 17.39
## vocaatis    7.31 10.45 -20.20 26.60 -8.30 -0.86 -6.89 15.33
## vocant     -1.46  7.22 -17.23 25.81 -4.29 -1.89 -0.27  9.12
## clamoo     -6.23 -3.66 -18.95 12.46 -6.20  7.27  8.42  5.25
## clamaas    -3.25 -4.53 -14.43 19.94 -6.85  4.02  3.84  3.77
```

constructing semantic vectors for complex words



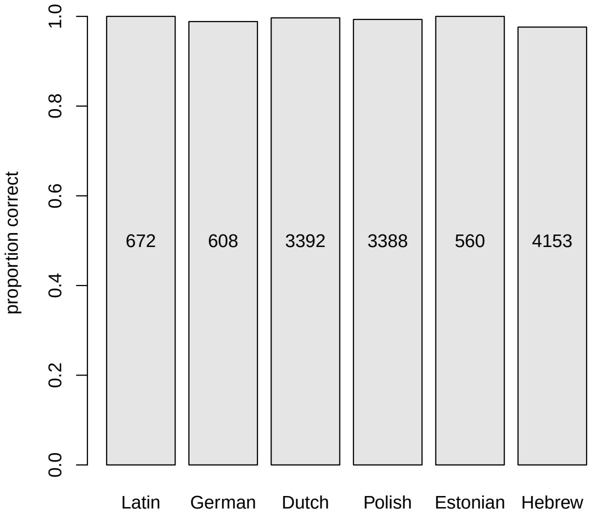
the semantic vector of an inflected word is defined as
the sum of the semantic vectors of its stem and affixal functions

evaluating comprehension accuracy



select as recognized the word with the highest correlation with the estimated semantic vector

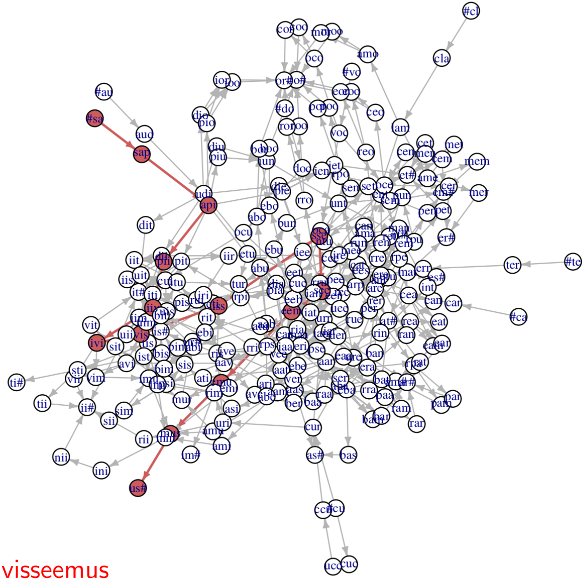
comprehension accuracy using simulated semantic vectors



evaluation production accuracy

- a semantic vector is mapped onto a form vector
- the form vector assigns a weight to each trigram/triphone, but by itself does not specify the ordering of these trigrams
- however, trigrams contain information on partial orderings thanks to their overlap: ABC $\rightarrow$ BCD, ABC $\nrightarrow$ XYZ
- this makes it possible to set up a directed graph with trigrams as vertices and edges between triphones that properly overlap

words are paths in a directed graph



sapiivisseemus

selecting the optimal path

- candidate paths are paths leading from an initial triphone (#AB) to a final triphone (XY#)
- select for articulation that path
  1. for which its corresponding estimated semantic vector is closest to the semantic vector to be articulated, and
  2. that has the smallest ratio R of path length to weakest link

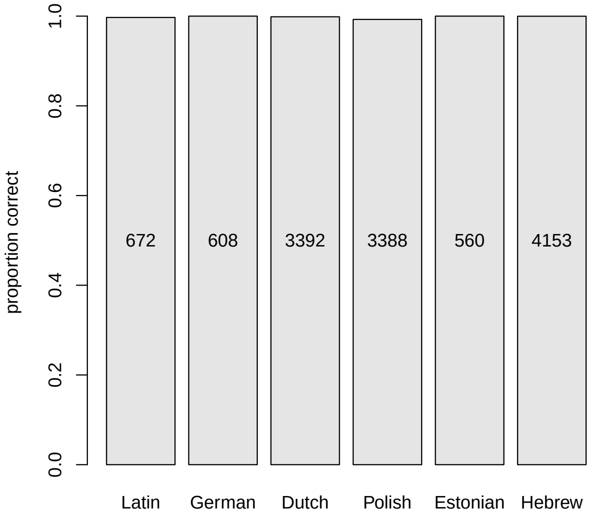
$$R = \frac{\#\text{vertices}}{\min(\boldsymbol{w})},$$

  where $\boldsymbol{w}$ is the vector of weights (activations) of all (non-initial) triphones in a candidate's path generated by the mapping from meaning to form

selecting the optimal path

- it sometimes happens that the algorithm finds a path that better expresses the semantic vector targeted for articulation than the form in actual use
- e.g., for our Latin dataset: curriaaris instead of curraaris (the form appropriate for the 4th conjugation class)

production accuracy for simulated semantic vectors



results for Biblical Hebrew

the Biblical Hebrew Corpus

- 271,299 words
- 19,339 unique inflected verb forms
- 23,834 unique nominal forms

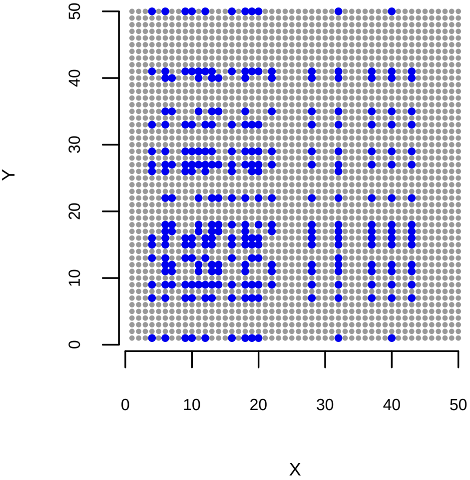
accuracy

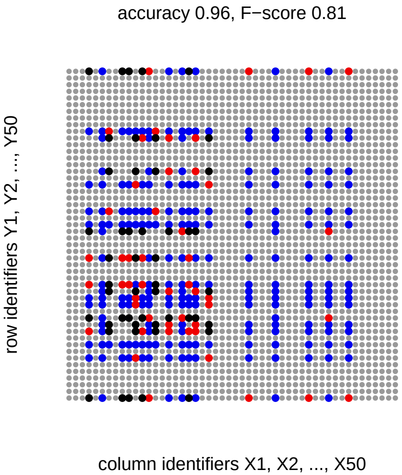
	comprehension		production	
	empirical	simulated	empirical	simulated
verbs	0.721	0.949	0.414	0.970
nouns	0.766	0.966	0.280	0.931
verbs and nouns		0.896		0.901

why does wide learning work so well?

- Minsky & Papert (1972): two-layer networks can only solve classification problems that are linearly separable
- so are 'language problems' basically simply linearly separable?
- or perhaps the claim of Minsky & Papert is too strong, given that it is based on a specific geometric approach to the problems they were interested in

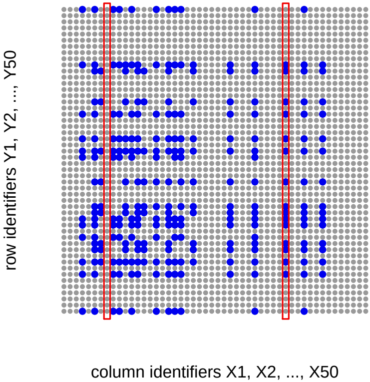
classification example, defined in a Cartesian plane

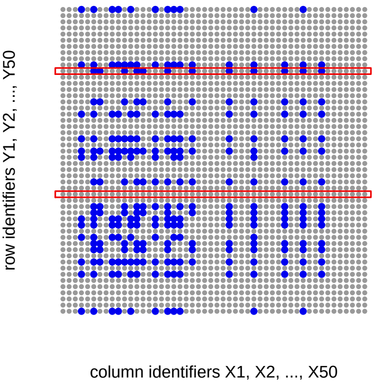


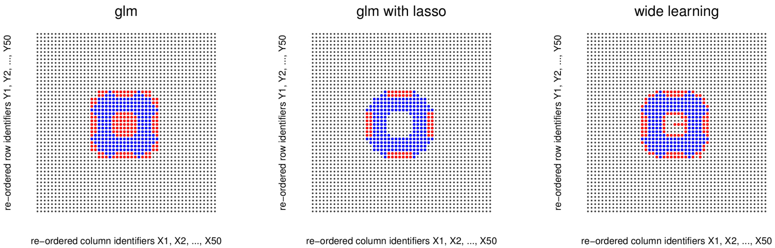


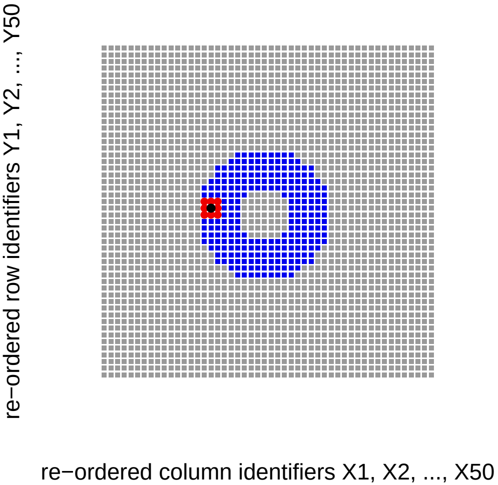
GLM: Accuracy 0.95, F-score 0.76

GLM with LASSO: Accuracy 0.98, F-score 0.87


row identifiers Y1, Y2, ..., Y50
column identifiers X1, X2, ..., X50


row identifiers Y1, Y2, ..., Y50
column identifiers X1, X2, ..., X50


glm
glm with lasso
wide learning
re−ordered row identifiers Y1, Y2, ..., Y50
re−ordered column identifiers X1, X2, ..., X50



cues: is hub, is neighbor of hub, is not a hub, is not a neighbor of a hub
100% accurate

deep learning and regression

Polynomial Regression As an Alternative to Neural Nets

Xi Cheng
Department of Computer Science
University of California, Davis
Davis, CA 95616, USA xicheng@ucdavis.edu

Bohdan Khomtchouk
Stanford University
Stanford, CA 94305, USA
bohdan@stanford.edu

Norman Matloff
University of California, Davis
Davis, CA 95616, USA
matloff@cs.ucdavis.edu

Pete Mohanty
Stanford University
Stanford, CA 94305, USA
pmohanty@stanford.edu

June 19, 2018

Abstract

Despite the success of neural networks (NNs), there is still a concern among many over their "black box" nature. Why do they work? Here we present a simple analytic argument that NNs are in fact essentially polynomial regression models. This view will have various implications for NNs, e.g. providing an explanation for why convergence problems arise in NNs, and it gives rough guidance on avoiding overfitting. In addition, we use this phenomenon to predict and confirm a multicollinearity property of NNs not previously reported in the literature. Most importantly, given this loose correspondence, one may choose to routinely use polynomial models

auditory comprehension revisited

- recent extensions and comparison with deep learning
- limitations of our approach

from NDL to NDL+

- a second network takes the output of the ndl network as input
- it is trained (by solving $\boldsymbol{AF} = \boldsymbol{T}$) to again predict the words (using vectors with one-hot encoding)

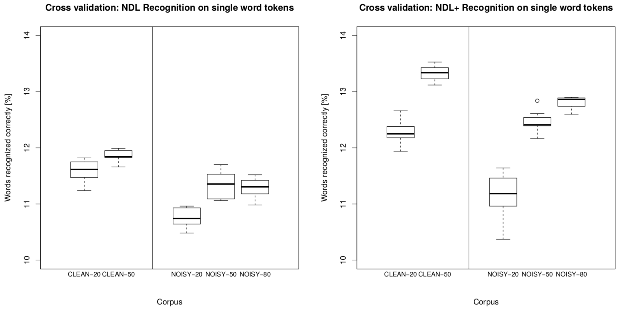
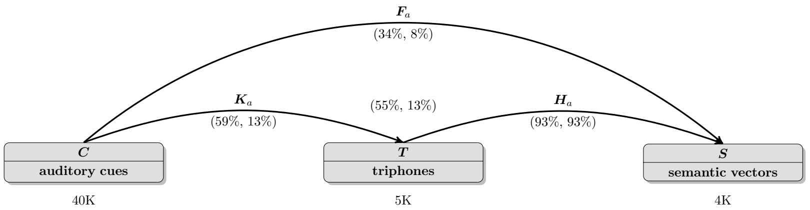


Figure 1: *Box-and-whiskers plots for the accuracy of word identification [%] across 10-fold cross-validation on five corpora for the NDL model in isolation (left panel) and for the NDL+ model paired with NDL (right panel).*

auditory word recognition with LDL

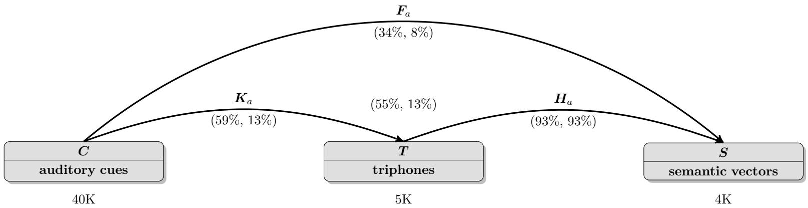
- route 1: map acoustic feature vectors directly onto semantic vectors (word embeddings)
- route 2: first map acoustic feature vectors onto triphone vectors, then map the triphone vectors onto semantic vectors



comparison with deep speech

- **isolated word recognition**
  - Mozilla Deep Speech: 6% correct on isolated word recognition (trained on thousands of hours of speech)
  - LDL indirect route: 55% (13% under 10-fold cross-validation) (10 hours of clean speech, American news broadcasts)
- **recognition of continuous speech**
  - Mozilla Deep Speech works impressively well
  - as yet unknown whether we can make LDL effective for continuous speech recognition

addressing overfitting



- dimension reduction of the matrix with acoustic cues
- replace linear mapping with LSTM ?
- hypercube-based topological coverings?

hypercube-based topological coverings

A Constructive Approach for One-Shot Training of Neural Networks Using Hypercube-Based Topological Coverings

W. Brent Daniel, Enoch Yeung

Abstract—**In this paper we presented a novel constructive approach for training deep neural networks using geometric approaches. We show that a topological covering can be used to define a class of distributed linear matrix inequalities, which in turn directly specify the shape and depth of a neural network architecture. The key insight is a fundamental relationship between linear matrix inequalities and their ability to bound the shape of data, and the rectified linear unit (ReLU) activation function employed in modern neural networks. We show that unit cover geometry and cover porosity are two design variables in cover-constructive learning that play a critical role in defining the complexity of the model and generalizability of the resulting neural network classifier. In the context of cover-constructive learning, these findings underscore the age old trade-off between model complexity and overfitting (as quantified by the number of elements in the data cover) and generalizability on test data. Finally, we benchmark on algorithm on the Iris, MNIST, and Wine dataset and show that the constructive algorithm is able to train a deep neural network classifier in one shot, achieving equal or superior levels of training and test classification accuracy with reduced training time.**

I. Introduction

Artificial neural networks have proven themselves to be useful, highly flexible tools for addressing many complex problems where first-principles solutions are infeasible, impractical, or undesirable. They have been used to address challenging classification problems ranging from wine typing to complex image analysis, voice recognition, language translation, and beyond.

The same flexibility that allows neural networks to be applied in such disparate contexts, however, can also lead to ambiguity in their appropriate definition and training. Deep neural networks, for example, are composed of multiple hidden layers with each hidden layer containing many nodes, each completely connected to the nodes in the preceding layer by a set of weights $\mathbf{W}_h$. Historically there has not been a functional relationship or algorithmic approach that allows researchers to define or derive a neural network's structural characteristics from either the problem specification or the associated training data. A neural network's topology *can* be optimized for a given problem, but this effectively results in a nested series of optimizations with the outer optimization steps tasked with incrementally assessing the most effective network topology [1].

Similarly, there has been no *a priori* way to specify input parameters. Many algorithms, especially those that rely on gradient information, can become stuck in local minima, limiting the predictive quality of a network for a given training instance. The result is that the same structural topology and training data can yield neural networks with a broad distribution of predictive qualities from one training run to the next. These stochastic effects can be most marked when the volume of training data is relatively small, yielding an optimization problem with relatively few constraints compared to the dimensionality of the parameter space. Such effects can be reduced by the choice of training algorithm, its parameterization, or by repeated training restarts, but this correspondingly increases the computational complexity and training time. Additionally, it's typically impossible to unambiguously specify a finite stop condition for training. This is the result of three factors: the training process is stochastic, the metric space has the potential for local minima, and the global minimum value is unknown beforehand.

In this paper we introduce a constructive method for the design of neural networks that rely on geometric representation of the data. The algorithm directly addresses the issues outlined above, including, 1) providing a concise structural definition of the neural network and its topology, 2) assigning network connection weights deterministically, 3) incorporating approximations that allow the algorithm to construct neural networks that in many cases have greater mean accuracy and better precision than traditionally trained networks, especially when training data is relatively sparse, 4) having a well-defined stop condition for training, and 5) inherently providing a clear interpretation of what and how information is encoded within the resulting neural network.

II. Constructive Learning Using Topological Covers

In what follows, we introduce a three-step approach for constructive training of a ReLU neural network:

- One or more topological covering maps are defined between a network's desired input and output spaces;
- These covering spaces are encoded as a series of linear matrix inequalities; and, finally,
- The series of linear matrix inequalities is translated into a neural network topology with corresponding connection weights.

R packages

- ndl
- acousticNDLCoder
- WpmWithLdl (to go on CRAN in a couple of months)