Informational Neurobayesian Approach

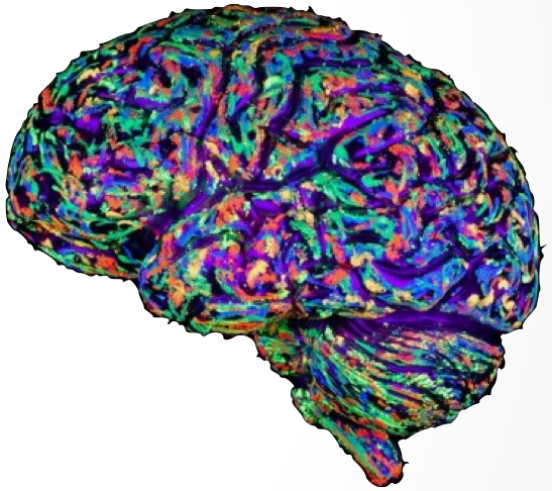
A. Artemov,
Cognitive Systems Company
Cogsys.company

10/11/2017

Problem Manifestation

Is brain just a large neural network?!

Human brain consists of ~100 bln. synapses with 125 trl. Parameters*, which is the biggest neural network. Today it is possible to build large neural networks, but what if there are many? For example the brain consists of 300 mln. neural networks which are interconnected with each other. How to do that?



DON FARRALL VIA GETTY IMAGES

* Azevedo et. all, 2017

Ideal Approach

An approach to build such large and powerful systems as the human brain should have the following characteristics (as minimum):

1. A large number of parameters with fast learning
2. High accuracy level (classification/forecast)
3. Universalism - combining or decomposing the features/classes of neural networks, rapid overfitting
4. “Humanity” - convenience of programming

Gradient Descent

Gradient Descent – most popular way of training neuromodels
Computational complexity – $O(n^2)$

$$\nabla Q(w_0, w_1, \ldots, w_n) = \left[\frac{\delta Q}{\delta w_0}, \ldots, \frac{\delta Q}{\delta w_n} \right]$$

So, on each iteration the weights are updated according to the following rule:

$$w_i = w_i - \mu \frac{\delta Q}{\delta w_i}$$

Main problem: requires too much computational power for large-scale models

Neurobayesian Approach

Computational complexity – O(n)

Advantage – neural network weights are calculated as probability

Drawback – does not take into consideration the data's structure and intrinsic features

$$\log p(X_{tr}|W) = \int q(T) \log p(X_{tr}|W) dT =$$

$$= \int q(T) \log \frac{p(X_{tr}, T|W)}{q(T)} dT + \int q(T) \log \frac{q(T)}{p(T|X_{tr}, W)} dT =$$

$$\mathcal{L}(q, W) + KL(q(T)||p(T|X_{tr}, W))$$

If only X_{tr} is known, T is a hidden variables vector, p(X), q(T) are their density functions respectively, and W was determined during optimization

EM-Algorithm

1. **E-step** $\mathscr{L}(q, W_{t-1}) \to \max_q$ - corresponds to KL-divergence minimization.

$$q_t(T) = \arg\min_q KL(q(T)||p(T|X_{tr}, W_{t-1})) = p(T|X_{tr}, W_{t-1})$$

2. **M-step** $\mathscr{L}(q, W_{t-1}) \to \max_W$.

$$W_t = \arg\max_w \mathscr{L}(q, W_{t-1}) =$$

$$= \arg\max_w \int q_t(T) \log \frac{p(X_{tr}, T|W)}{q_t(T)} dT =$$

$$= \arg\max_w \int q_t(T) \log p(X_{tr}, T|W) dT,$$

which is a concave function, so the maximisation is legal.

The procedure is repeated until convergence.

Methods for Building Large Neural Networks

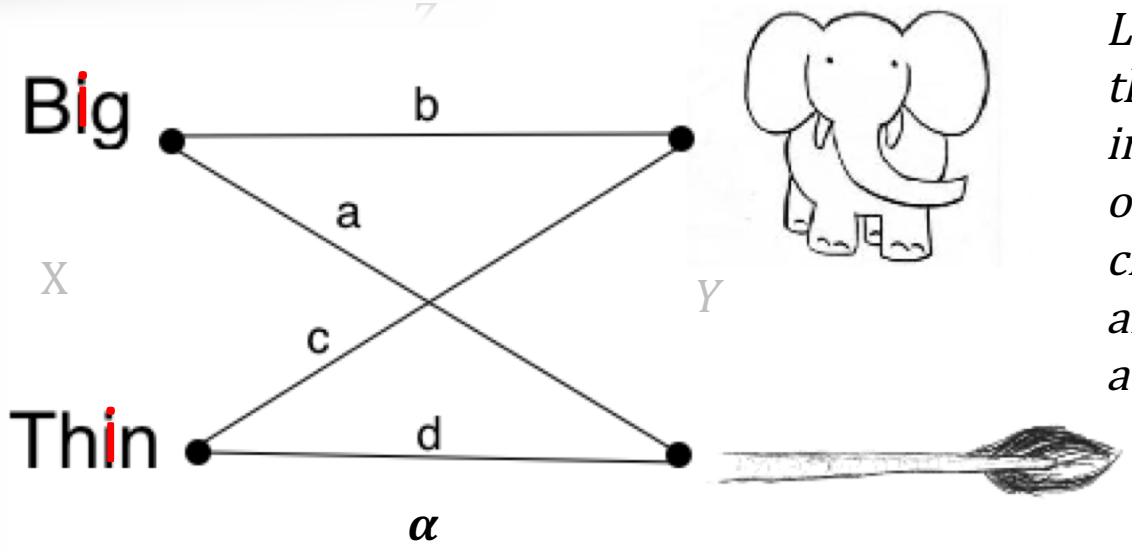
Learn by method on 1 Tflops CPU	How long to train a model with 125 trln synapses	Verification
Back-propagation (gradient descent)	49 years	Google experiment results * estimation (4,3 bln params for 17 hours on 1,2 TFlops GPU)
Neurobayesian (EM-algorithm)	no information	no information
INA Neurobayesian (EM-algorithm)	< 1 year	Cogsys Company experiment results estimation (5 bln. params for 2,5 hours on 0,2 TFlops CPU)

Biggest language neuromodel today – 137 bln. parameters (synapses)

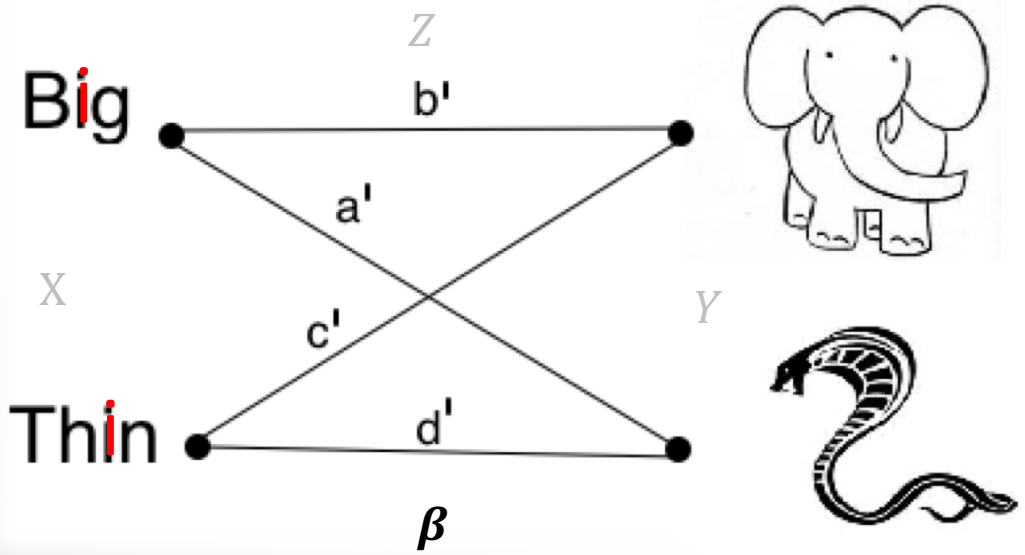
How to build larger neural networks?

* Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538

Blind Men, a Snake and an Elephant Task



$\boldsymbol{\alpha}$



$\boldsymbol{\beta}$

Let there be $\boldsymbol{\alpha}$ *and* $\boldsymbol{\beta}$ *neural networks trained to classify the feature perceptions of blind people on different initial data. And we would like to make a choice based on the forecast of the two networks. How to choose a class on two neural networks: is it an elephant or not an elephant? (how to add the vectors of weights in the absence of a single basis).*

$$b > d, c' > a'$$
$$b + c' > ?$$

The lack of interpretations of synapses weight *coefficients in artificial neural networks makes it incorrect to conduct algebraic operations with them as identical values of weights in different models are not equivalent.*
This is similar to the «mathematics» of the Nivhi tribe - for round objects they have one set of numbers, for elongated - another.

The simple solution is the function $\phi(x, y) = \phi(x) \cdot \phi(z) \cdot \phi(y)$

$$\forall \alpha, \beta \ \exists \ \phi(z_\alpha) = \phi(z_\beta) \leftarrow \frac{\phi(x_\alpha, z_\alpha)}{\phi(x_\alpha)\phi(y_\alpha)} = \frac{\phi(x_\beta, z_\beta)}{\phi(x_\beta)\phi(y_\beta)}$$

$$\Rightarrow \sum_\alpha \phi(z_\alpha) = \sum_\beta \phi(z_\beta)$$

$$\Rightarrow \sum_\alpha \frac{\phi(x_\alpha, z_\alpha)}{\phi(x_\alpha)\phi(y_\alpha)} = \sum_\beta \frac{\phi(x_\beta, z_\beta)}{\phi(x_\beta)\phi(y_\beta)}$$

Methods for Building Large Neural Networks

Approach	Weights Interpretation	Additivity
Back-propagation (gradient descent)	no	No (Not correct)
Neurobayesian (EM-algorithm)	probabilistic	?
INA Neurobayesian (EM-algorithm)	quantity of information	yes

How to maintain additivity without increasing complexity?

The Bit Story

The number of messages *N* that can be obtained by combining *m* alphabet characters (unique) by *k* alphabet characters (all) in the message is:

$$N = m^k$$

Hartley proposed to measure the quantity of information as a logarithm of the number of possible sequences of symbols:

$$I = \log N = \log m^k = k * \log m$$

$$N = m^k = 2^I$$

If we have 10 features (including all possible forms) you can encode no more than 2^W-1 combinations. It means that the maximum number of classes encoded by these features, where each feature carries no more than $I=\log_2 2^w-1 = 9.9$ bits of information. And on the other hand if you know that a feature transmits X quantity of information about class Y - it means, that there were 2^I precedents about this particular feature initiating the respectable class. Convenient!

Quantity of Information

For the compilation of the message let there be k symbols possessing *m* qualitative features and p_i - probability of each qualitative feature or symbol. Shannon designed a method for determining the average amount of information in a message with arbitrary probability of symbol values:

$$I = -k \sum_{i-1}^{m} p_i \log p_i$$

In information theory, entropy is the measure of uncertainty. For a message from k elements, it can be calculated as:

$$H = \frac{I}{k} = -\sum_{i-1}^{m} p_i \log p_i$$

Often, when solving problems, researchers tend to forget about K coefficient, because it is necessary to determine functional dependency for measuring the message's length or to fix the message's length, which in essence is equivalent to k=1

Rich Bayes

$$P(w|m) = \frac{P(w;m)}{P(m)} = \frac{P(m|w)P(w)}{P(m)}$$

$$\lambda(w;m) \equiv \frac{P(w|m)}{P(w)} = \frac{P(w;m)}{P(w)P(m)} = \frac{P(m|w)}{P(m)}$$

$$\lambda(w_i;m) = \frac{P(w_i;m)}{P(w_i)\sum_{j=1}^{W} P(m|w_j)P(w_j)} = \frac{P(m|w_i)}{\sum_{j=1}^{W} P(m|w_j)P(w_j)}$$

What does it provide us with? Only with the link between the events in the coefficient $\lambda(w;m)$. It tells us by how many times the probability of event ***m*** increases when observing event ***w*** and vice versa.

$$\lambda(w_i;m)P(w_i) = \frac{P(m|w_i)P(w_i)}{\sum_{j=1}^{W} P(m|w_j)P(w_j)}$$

Naive Bayes

$$\lambda(w_i;w_k) = \frac{P(w_i;m_k)}{P(w_i)P(m_k)}$$

Looks like required weight $\phi(z)$
but the sum $\phi(z)$ is not interpretable
And $\prod \phi(z_i)$ - is bad way in case $\phi(z_j)$=0

Example

We have 10 000 receipts of purchases. In 6 000 receipts there is a purchase of computer games, in 7 500 receipts there is a purchase of videos; in 4000 receipts there is a purchase of video and computer games simultaneously. So, we estimate the probability.

The probability of buying a game:
P (game) = 6000/10000 = 0.6 (60%)

Probability of buying a video:
P (video) = 7500/10000 = 0.75 (75%)

The probability of buying a game, provided that the receipt has a purchase of video:
P (game | video) = 4000/7500 = 0.533 (53.3%)

The probability of buying a video, provided that the receipts has a purchase of a computer game:
P (video | game) = 4000/6000 = 0.667 (66.7%)

As we see P (game)> P (game | video), similarly to P (video)> P (video | game). That means, the observation of one of the events reduces the probability of observing another one. We can even count by how much:
lambda (game, video) = P (game | video) / P (game) = 0.889 = P (video | game) / P (video)

*We thank Denis Astanin for this example (habrhabr)

Bayes Theorem + Bit = PMI

We can rewrite $\lambda(\mathrm{w};\mathrm{m})$ in the terms of quantity of information:

$$\mathrm{PMI}(w_i;\mathrm{m}) = \log\frac{\mathrm{P}(w_i;\mathrm{m})}{\mathrm{P}(w_i)\sum_{j=1}^{W}\mathrm{P}(\mathrm{m}|w_j)\mathrm{P}(w_j)} = \log\frac{\mathrm{P}(\mathrm{m}|w_i)}{\sum_{j=1}^{W}\mathrm{P}(\mathrm{m}|w_j)\mathrm{P}(w_j)}$$

- Unrelated events - $\lambda(\mathrm{w};\mathrm{m}) = 1$;
- Positive correlation - $\lambda > 1 \Rightarrow PMI > 0$;
- Negative correlation - $\lambda < 1 \Rightarrow PMI < 0$

$\mathrm{h}(w_i) + \mathrm{h}(m_k) - \mathrm{h}(w_i; w_k) = \mathrm{PMI}(w_i; w_k)$
Required additive weight $\phi(\mathrm{z}) = \mathrm{PMI}(w_i; w_k)$
sum $\phi(\mathrm{z})$ - is interpreted as a total measure of additional information entropy

Thus, PMI is an indicator of the degree of relationship between events:

$$-\infty \le PMI \le \min[h(w); h(m)]$$

It is also possible to ensure the normalization condition [-1; 1]

$$NPMI(w_i;\mathrm{m}) = \frac{PMI(w_i;\mathrm{m})}{h(w_i;\mathrm{m})} = \frac{\log P(w_i)P(m)}{\log \mathrm{P}(w_i;\mathrm{m})} - 1$$

no co-occurrences,log P(m, w_i)→-∞ so *nmpi* is -1,
co-occurrences at random, log (p, w_i) =log$P(w_i)P(m)$ so *nmpi* is 0,
complete co-occurrences, *log* $P(w_i)$= *log* $P(m, w_i)$=log$P(m)$= so *nmpi is 1.*

Weighted mutual information

$$I(\mathrm{W};\mathrm{M}) = H(W) + H(M) - \mathrm{H}(W;M) = \sum_W P(w) \sum_M P(w|m) \log \frac{\mathrm{P(W|m)}}{\mathrm{P(w)}} = \sum_B P(w) D_{KL}\big(P(w,m)||P(m)\big) = \mathbb{E}_M\big[D_{KL}\big(P(w,m)||P(m)\big)\big]$$

$$wI(\mathrm{W};\mathrm{M};\mathrm{Z}) = \sum_W \sum_M Z(w,m) P(w;m) PMI(\mathrm{w};\mathrm{m})$$

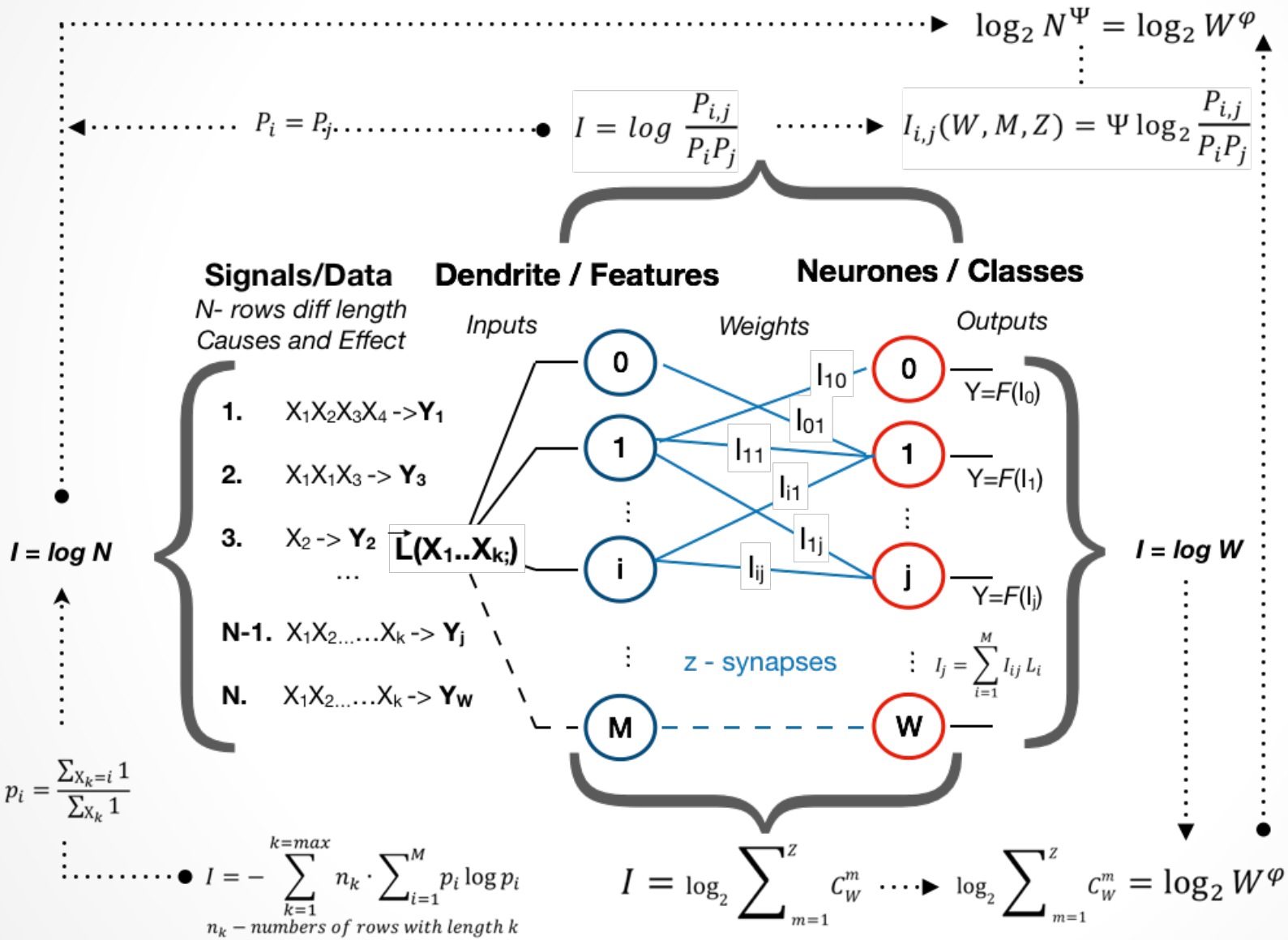
which places a weight Z(w,m) on the probability of each variable value co-occurrence p(w,m). This weighted mutual information is a form of weighted KL-Divergence, which is known to take negative values for some inputs. We will use this to weight the influence of the different input samples.

$$wN\_I(w_i;\mathrm{m},\mathrm{z}) = \frac{\sum_{w_i} \sum_m \overbrace{Z(w_i,m) * P(w_i;\mathrm{m})}^{\to \delta} * PMI(w_i;\mathrm{m})}{h(w_i;m)} \leq \frac{\delta * W_i}{h(w_i;m)} \sum_m PMI(w_i;\mathrm{m}) \approx$$

$$\approx \frac{\varphi \log W_i}{\log N_i} \sum_m PMI(w_i;\mathrm{m}) = \Psi * \sum_m PMI(w_i;\mathrm{m}) = \frac{\log \sum_{m=1}^{Z} C_W^m}{\log N_i} \sum_m PMI(w_i;\mathrm{m})$$

$Z_i = W_i * \mathrm{M}$ – numbers of synapses, $N - \ occurences\ \mathrm{m} \to w_i$ W_i - numbers of classes w_i , M – numbers of features. φ, Ψ - coefficient of emergence of classes and features, respectively. $Z * P * N\_PMI \in [-\delta;\delta]$, $\left\{\delta \leq \left|\frac{1}{Z(w_i,m)} \pm P(w_i;\mathrm{m})\right| |P(w_i;\mathrm{m}) \to \frac{1}{Z(w_i,m)}\right\}$. Notice that optimum of $Z(w_i,m) = \frac{1}{2} + \sqrt{1 + \frac{1}{P(w_i;\mathrm{m})}}$. Z is a function that determines the length ($\mathrm{z} \in \mathbb{N}$) of the message - a set of features in the object. $\delta * W_i$ is close but not equivalent to the number of synapses that connect the feature and classes (neurons). Variance of δ sets the noise level in the model.

System generalization



$$I(W;M) =$$

$$= \log M + Log\ W - Log\ Z$$

$$= \log M * W - Log\ Z$$

$$= \Psi * \log N - Log\ Z$$

$$= \varphi * \log \mathrm{W} - Log\ Z$$

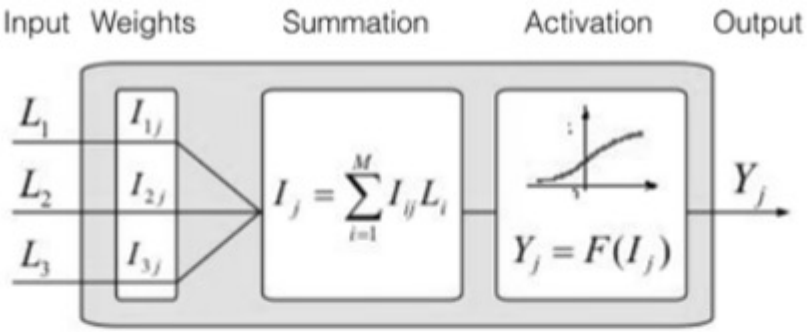
$$= \log \sum_{m=1}^{Z} C_W^m - Log\ Z$$

$$= \sum_k \left(\log \sum_{m=1}^{W_k} C_W^m - Log\ W_i\right)$$

$$= \sum_k i_k(w\ ;\ m)$$

Summation Process

$$I_{i,j} = \Psi * \log_2 \frac{P_{i,j}}{P_i P_j}$$



Summation Process Architecture

In the scheme above the L_i is i-th object's feature. $\vec{L}\{X_1, \ldots, X_k\}$– a vector of object features:

$$L_i = \begin{cases} 1, & \text{if } X_k \geq \varepsilon \\ 0, & \text{otherwise} \end{cases}$$

For every object with a given set of features the neural network chooses over all classes the one with maximal value of the activation function, represented as an activated sum by all features.

Informational Neurobayesian Approach Emergence and Entropy in Neural Networks

Computational complexity – O(nlog$n$) (E-Step – O(n), M-Step – O(log$n$)

Advantage – weights as the amount of information – additivity property

General formula of Lutsenko for Harkevich emergence coefficient:

$$\psi = \frac{\log_2 W^{\varphi}}{\log_2 N}$$

where $\varphi = \frac{\log_2 \sum_{m=1}^{Z} C_W^m}{\log_2 W}$.

ψ varies from 0 to 1 and determines the degree of determination of the system

Problem – how to determine φ *means:*

$$\boldsymbol{\psi_{min}} = \frac{\log_2 W}{\log_2 N}$$

$$\boldsymbol{\psi_{real}} = \frac{\log_2 \sum_m C_W^m}{\log_2 N}$$

$$\boldsymbol{\psi_{max}} = \frac{\log_2 (2^{W_i} - 1)}{\log_2 N_i}$$

Emergence effect in this context can be explained in the following way: with one symbol (feature) it is possible to encode 1 class, emerg= 1/1=1, with 2 features – 3 classes, emerg = 3/2, with 3 – 7, emerg = 7/3 etc. The final formula for emergence coefficient for coding class with features for N- occurrences

$$\boldsymbol{\psi} = \frac{\log \sum number\ of combinations\ feature\&class}{\log Number\ of\ occurences\ feature\&class}$$

Summation Process

$$P_{i,j} = \frac{N_{i,j}}{N};\ P_i = \frac{N_i}{N};\ P_j = \frac{N_j}{N}$$

$$I_{i,j} = \Psi * \log_2 \frac{N_{i,j} N}{N_i N_j}$$

		Classes					Sum
		1	...	j	...	W	
Features	1	N_{11}		N_{1j}		N_{1W}	
	...						
	i	N_{i1}		N_{ij}		N_{iW}	$N_i = \sum_{j=1}^{W} N_{ij}$
	...						
	M	N_{M1}		N_{Mj}		N_{MW}	
Total observations				$N_j = \sum_{i=1}^{M} N_{ij}$			$N = \sum_{i=1}^{W} \sum_{j=1}^{M} N_{ij}$

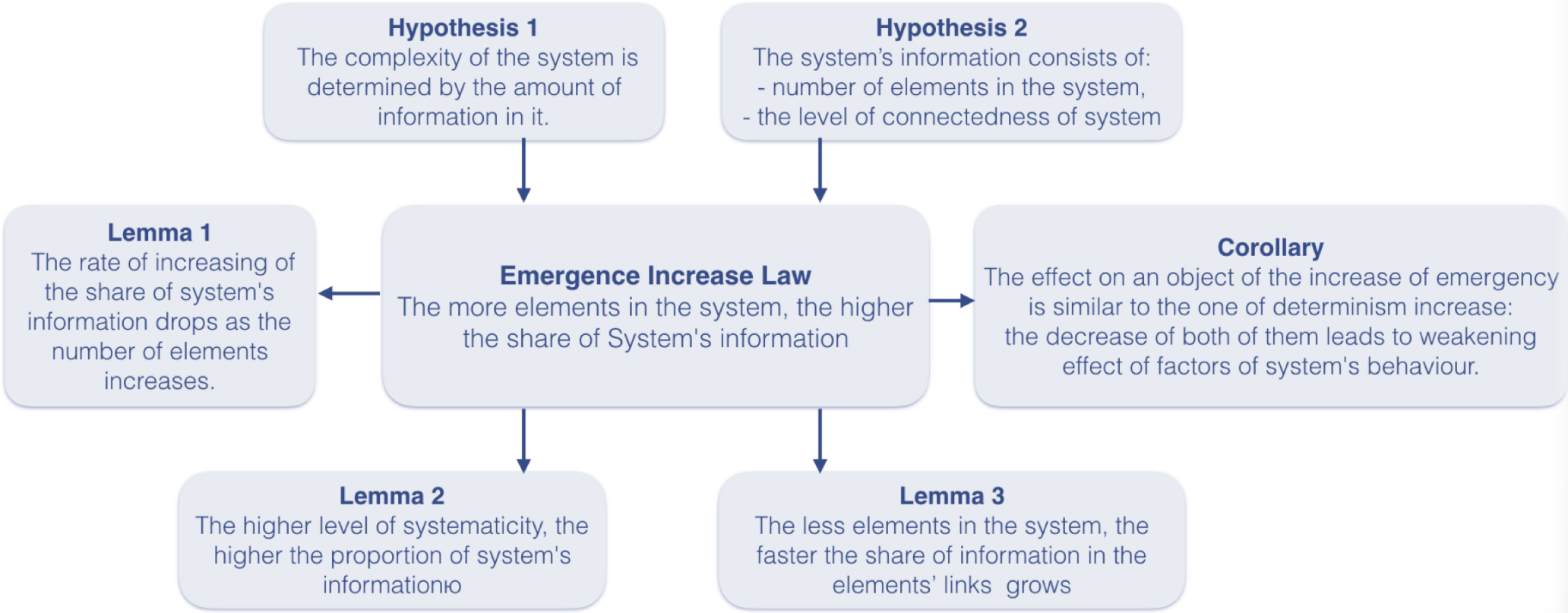
$$S_j = \sum_{j=1}^{M} I_{ij} \cdot L_i + \text{bias},\ \text{bias} = I_0$$

$$j^* = \arg\max_j f(S_j)$$

$$y = f(S_{j*})$$

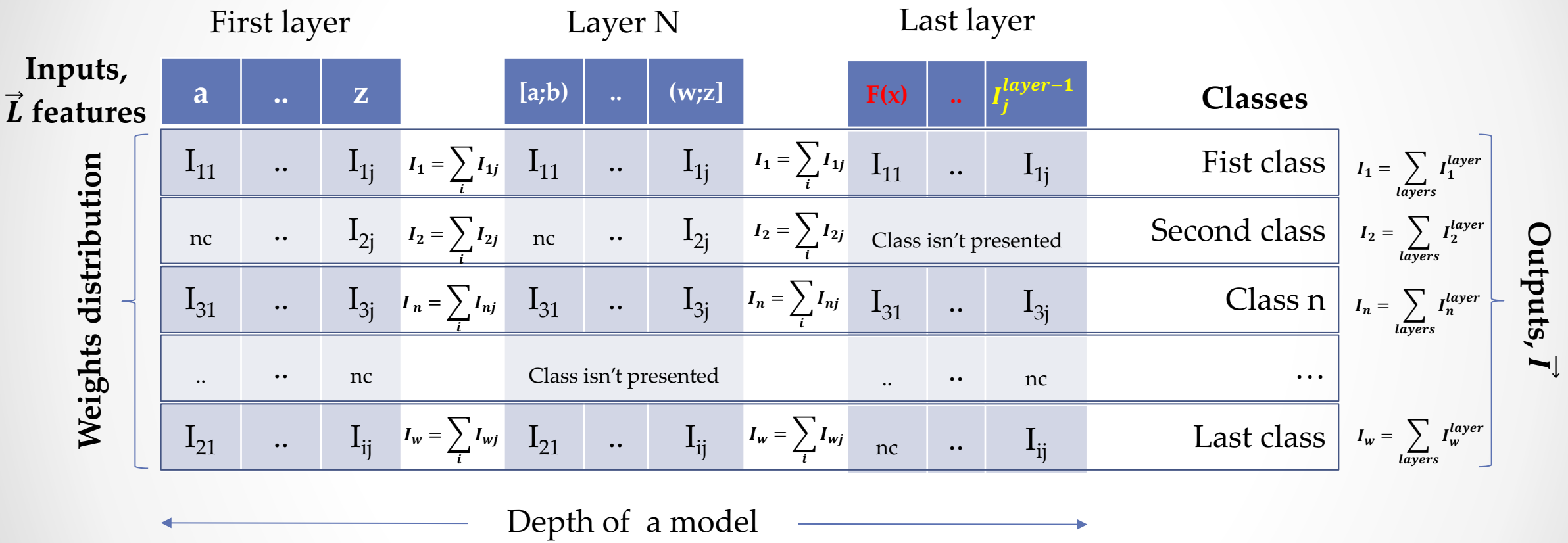
The Y is a chosen class for the given features, f is activation function.

Informational Neurobayesian Approach



Architecture of networks

A model example



* **nc -** no connection

Categorical, numeric values, quantity information from other model/layers and even functions, determining the values of variables. Layers of arbitrary models can be united in one model. In INA it is possible to add up weights attributing to one class.

One model for two types of work scenario

Parallel

First layer			Layer N			Last layer			Classes	
a	..	z	a	..	z	aa	..	zx		
I_{11}	..	I_{1j}	I_{11}	..	I_{1j}	I_{11}	..	I_{1j}	Fist class	I_1
nc	..	I_{2j}	nc	..	I_{2j}	I_{31}	..	I_{2j}	Second class	I_2
I_{31}	..	I_{3j}	I_{31}	..	I_{3j}	I_{31}	..	nc	Class n	I_n
..	..	..	..	..	..	..	..	..	…	
I_{21}	..	I_{ij}	I_{21}	..	I_{ij}	nc	..	I_{ij}	Last class	I_W

A wide variety

$$\begin{cases} I_1 = \text{PMI(a)}\ldots + \text{PMI(F(x))}\ldots + \text{PMI}\ (I_j^{layer-1}) \\ \ldots \\ I_w = \text{PMI(z)}\ldots + \text{PMI(F(x))}\ldots + \text{PMI}\ (I_j^{layer-1}) \end{cases}$$

the number of equations is less than the number of variables

$$J = \underset{j}{\operatorname{argmax}}\, AF(I_j)$$

Sequential (convolution on the fly)

First layer				Layer N				Last layer			Classes	
a	..	z		a	..	z		aa	..	zx		
I_{11}	..	I_{1j}	I_1	I_{11}	..	I_{1j}	I_1	I_{11}	..	I_{1j}	Fist class	I_1
nc	..	I_{2j}	I_2	nc	..	I_{2j}	I_2	I_{31}	..	I_{2j}	Second class	I_2
I_{31}	..	I_{3j}	I_n	I_{31}	..	I_{3j}	I_n					
..	..	..										
I_{21}	..	I_{ij}	I_W									

$$\vec{J} = \{AF(I_j) > \delta\}$$

$$J = \underset{j}{\operatorname{argmax}}\, AF(I_j)$$

$$\vec{J} = \{AF(I_j) > \delta\}$$

AF - activation function

Combining both scenarios, it is possible to create any kind of neural network architecture even from layers of different models

Dynamic Modeling & Cognitive operator

		classes					
		1	..	3	4	..	W
features	a	N_{11}	..	N_{1x}	I_{1x}	..	I_{1j}
	b	nc	..	nc	I_{2x}	..	I_{2j}
	..	N_{31}	..	N_{3x}	I_{3x}	..	I_{3j}
	..	..	..	..	nc	..	nc
	M	N_{xj}	..	N_{xj}	I_{xj}	..	I_{ij}

$$I_{ij} = \boldsymbol{PMI}(N_{xj} \ldots N_{\boldsymbol{ij}})$$

PMI is a cognitive operator that extracts information from data. Data is presented by the number of N cause-effect pairs.

This approach makes it possible to quickly and adaptively train knowledge models

EM-Algorithm with F-measure Maximization

- Step 0 - Calculate a frequency matrix for input data.

- Step E – Creating informational neurobayesian model

- Model maximization by Van Rijsbergen's effectiveness measure (F-measure)

$$I_{i,j}(W,M,Z) = \Psi_g * \log_2 \frac{N_{i,j} * N_g}{N_i N_j}$$

$$\Psi_g = \frac{\log_2(2^{Z_g}-1)}{\log_2 N_g}$$

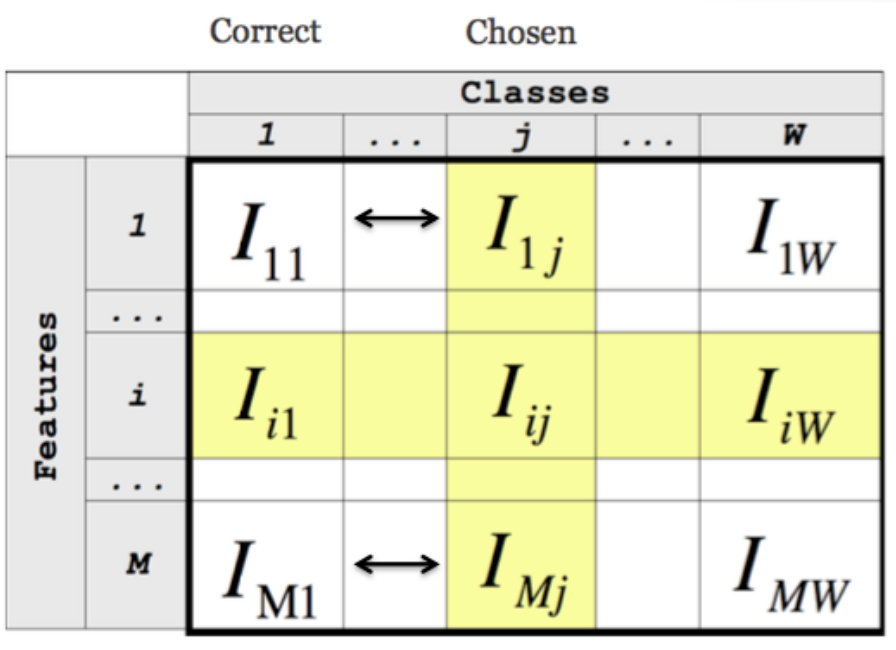
$$I_{i,j}(W,M,Z) \to \max_Z.$$

- Determine jt^* and jl^* - correct class for the set of features Li, $i \in M$

Getting rid of all the contradictive precedents

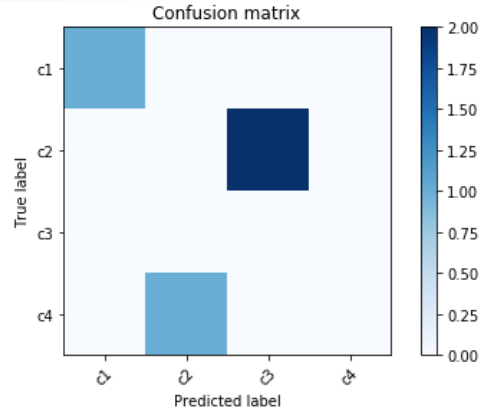
Fast Learning Trick

Will interchange the weights of the well-posed and ill-defined classes until the mathematical expectation of the micro F1-measure increases



Fast Learning Trick

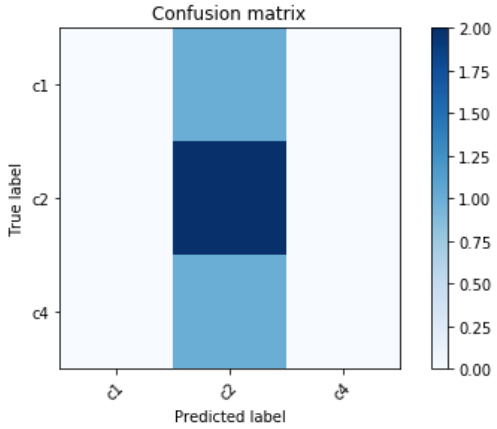
Start



Mean F-SCORE 0.25

	c1	c2	c3	c4
p1	1.06	0.70	0.00	1.64
p2	2.11	0.00	1.85	0.49
p3	0.44	1.76	1.93	0.41
p4	1.06	1.06	0.93	0.99

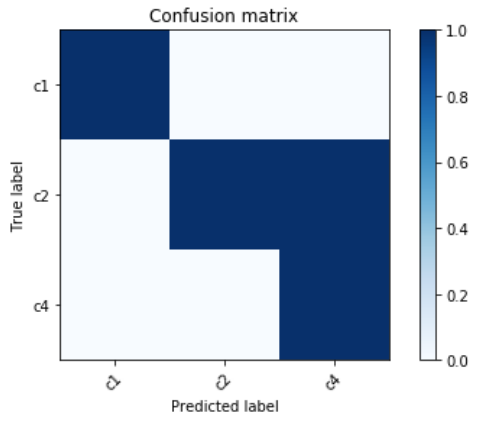
Epoch 1



Mean F-SCORE 0.666666666667

	c1	c2	c3	c4
p1	1.64	1.06	0.00	0.70
p2	1.85	2.11	0.00	0.49
p3	0.44	0.41	1.76	1.93
p4	1.06	1.06	0.93	0.99

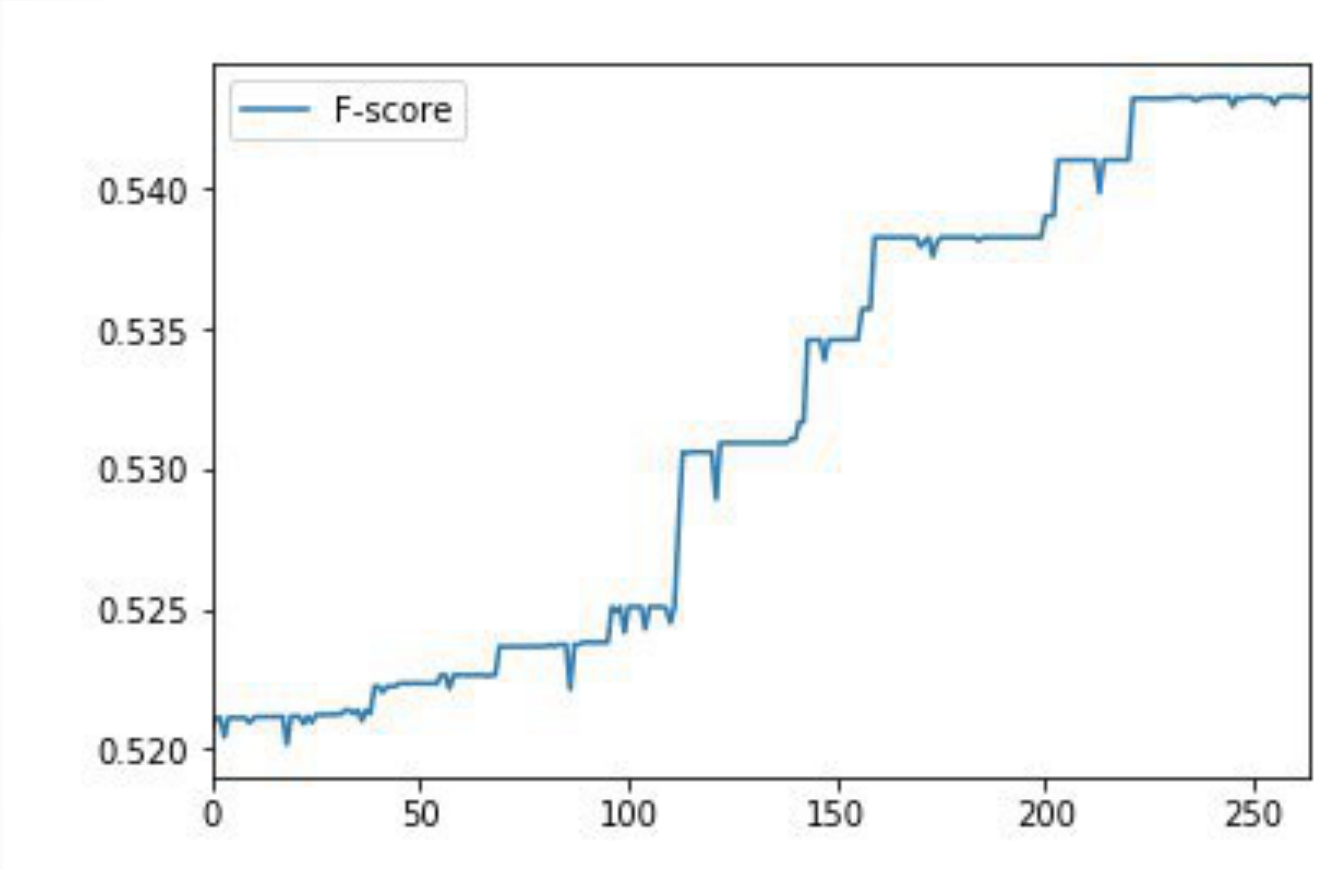
Epoch 2



Mean F-SCORE 0.75

	c1	c2	c3	c4
p1	1.64	1.06	0.00	0.70
p2	1.85	2.11	0.00	0.49
p3	0.44	1.93	1.76	0.41
p4	1.06	1.06	0.93	0.99

Fast Learning Trick



INA framework

Brain2

To use the INA in practice it was necessary to create our own framework, because none of the existing ones today (TensorFlow, Torch, Keras) is able to realise the proposed approach.

We are developing a framework called Brain2. And, accordingly, the solutions created on its base are named Brain2NAME (Brain2Text, Brain2Word and etc)

Achieved Results

Model	MNIST	House Prices	Emotions in text	Russian words	Russian words
Problem	Hand-written digits recognition	Real Estate Valuation	Determining emotions (by Ecman), in text.	Word recognition by letters and bigramms.	
Features	2 815	809	19 121	1042	1042
Classes	11	658	8	5 099 300	5 099 300
Parameters (weights)	30 965	532 322	152 968	6 313 470 600	6 313 470 600
Model Size (Mb)	-	1,89		634,94	634,94
Training data size (Mb)	74,9	0,53	2,85	505,43	505,43
Training data, rows	42 000	1460	8 682	5 099 300	5 099 301
Learning Type (1 epoch)	E	E	E	E	EM
Brain2 v.[a]	1.0	2.0	2.0	3.0	3.0
Result	CA [b]=0,81	RMSE=0,42	F1 = 0,81	F1=0,97	F1=0,98[c]
Time [d]	20 min	2 min 33 sec	6 min 4 sec	2 h, 35 min	2 h, 49 min

a Framework's version, developed by project's team, including paper's author

b This competition is evaluated on the categorisation accuracy of your predictions (the percentage of images you get correct).

***c* Perplexity of 1.26 per letter and bigrams (feature)**

d Tests were run on the machine with specifications: Intel Xeon, E5-1650 v3 Hexa-Core Haswell 6cores, 128 GB ECC RAM, 2 x 240 GB 6 Gb/s SATA SSD , 2X2Tb SATA3. GPUs were not used for training.

Best Model Selection

We have 13 models, each with 3 type of activation function – a total of 39 possibilities to solve 9 problems:

1) Naming
2) Apprehension
3) Generalization (induction)
4) Abstraction
5) Model adequacy assessment
6) Comparison and prognosis
7) Deduction and abduction
8) Classification
9) Planning

	I	norm_i	sr_i
model_1	1.000000	0.650433	0.916747
model_10	0.997986	0.478979	0.972803
model_11	0.956565	0.954717	0.483827
model_12	0.997986	0.623266	0.902697
model_13	0.993970	0.482042	0.958457
model_2	0.995976	0.479444	0.973440
model_3	0.916585	0.609586	0.957053
model_4	0.271126	0.834430	0.877937
model_5	0.949175	0.787008	0.808723
model_6	0.997986	0.937649	1.000000
model_7	0.446412	0.680246	0.997986
model_8	1.000000	0.613646	0.926647
model_9	0.997986	0.460722	0.945698

Correct word recognition

	I	norm_i	sr_i
model_1	0.390680	0.128911	0.338630
model_10	0.349639	0.079106	0.198291
model_11	0.316916	0.490791	0.281685
model_12	0.280278	0.114141	0.307991
model_13	0.357854	0.092771	0.291355
model_2	0.465080	0.108084	0.325444
model_3	0.066051	0.578422	0.254802
model_4	0.269745	0.138264	0.208901
model_5	0.317150	0.146170	0.210061
model_6	0.509749	0.640725	0.281768
model_7	0.305432	0.535861	0.271373
model_8	0.308524	0.167244	0.380397
model_9	0.355123	0.131978	0.312055

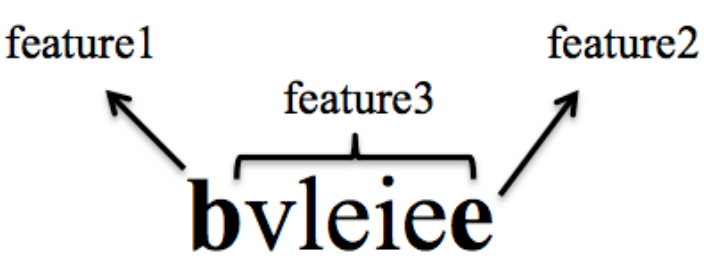
identifying lexico-grammatical word properties

Humanity Programming

How to choose model features in word recognition?

Example:

"*I cnduo't **b**vleie**e** taht I culod aulaclty uesdtannrd waht I was rdnaieg.*"



Which features are needed for a human to recognize the word *believe* in the symbols sequence *bvleiee?* First letter, last letter, the combination of letters in the center and length of the sequence.

You can train the neural network to reason like you (Turing test 2.0)

Equivalent - you understand each other

Spelling Error Models

Group	Algorithm name or author	Words in corpus	Incorrect Words in sample	Accuracy on words		Year	Source
				Only with errors	All types		
English words	Dronen	120 000	4 349		**0,92**	2016	1
	Hodge, Austin	81 717	600		**0,93**	2003	3
	Wikipedia ≥ 10	42 622	1100	**0,78**		2013	9
	Wikipedia ≥ 8			**0,75**		2013	9
	Wikipedia ≥ 6			**0,71**		2013	9
	Wikipedia ≥ 4			**0,68**		2013	9
	Ahmad, Kondrak	580 000	508		**0,79**	2005	6
Russian words	**Brain2Word**	**118 637**	**1) 25 000 2) 660**	**0,90**	**0,71***	**2017**	-
	MS Word 2007	-	56 000 Not publicly available	**0,88**		2015	8
	Yspell			**0,88**		2015	8
	Yandex			**0,84**		2015	8
	Google			**0,80**		2015	8
	CIGR Lemming			**0,80**		2015	8
	Aspell			**0,70**		2015	8
	MSU*	10 000 for tune up + dictionaries	1100 publicly available		**0,70**	2016	4
	Orfogrammatika*				**0,61**	2016	4
	MIPT*				**0,58**	2016	4
	ISP RAS*				**0,56**	2016	4
	HSE*				**0,50**	2016	4
	NLP@Cloud*				**0,34**	2016	5
	InfoQubes*				**0,25**	2016	4
	Dereza et al.	900 000	-		**0,40**	2016	8

* The system at this stage allows to correct spelling errors and typos.

Spelling Error Models

The presentation contains a table with the results of Brain2SpellCheck service. It was tested on two data sets:

1) 25 thousand words were selected from a set of 60 thousand words based on most common and relevant writing errors of people - from the research of D. Kulagin Word Map: Rethinking the Approach to Compiling Online Dictionaries in the Postmobile Era. Computer linguistics and intellectual technologies: proceedings of the international conference "Dialogue 2017", May 31 - June 3, 2017. http://cogsys.company/data/test_25k_errs.csv

2) The initial set of 1100 words with errors contained various kinds of generated mistakes – two words without spaces, scrambled letters etc. We used the dataset of 1100 words with errors only with orthographical mistakes and typos - proceedings of the Spellrueval Competition: the First Competition on Automatic Spelling Correction for Russian. Sorokin A. A., Baytin A.V., Galinskaya I.E., Shavrina T.O. "Dialogue-2016". June 1-4, 2016 http://cogsys.company/data/test_1200w_60-40.csv

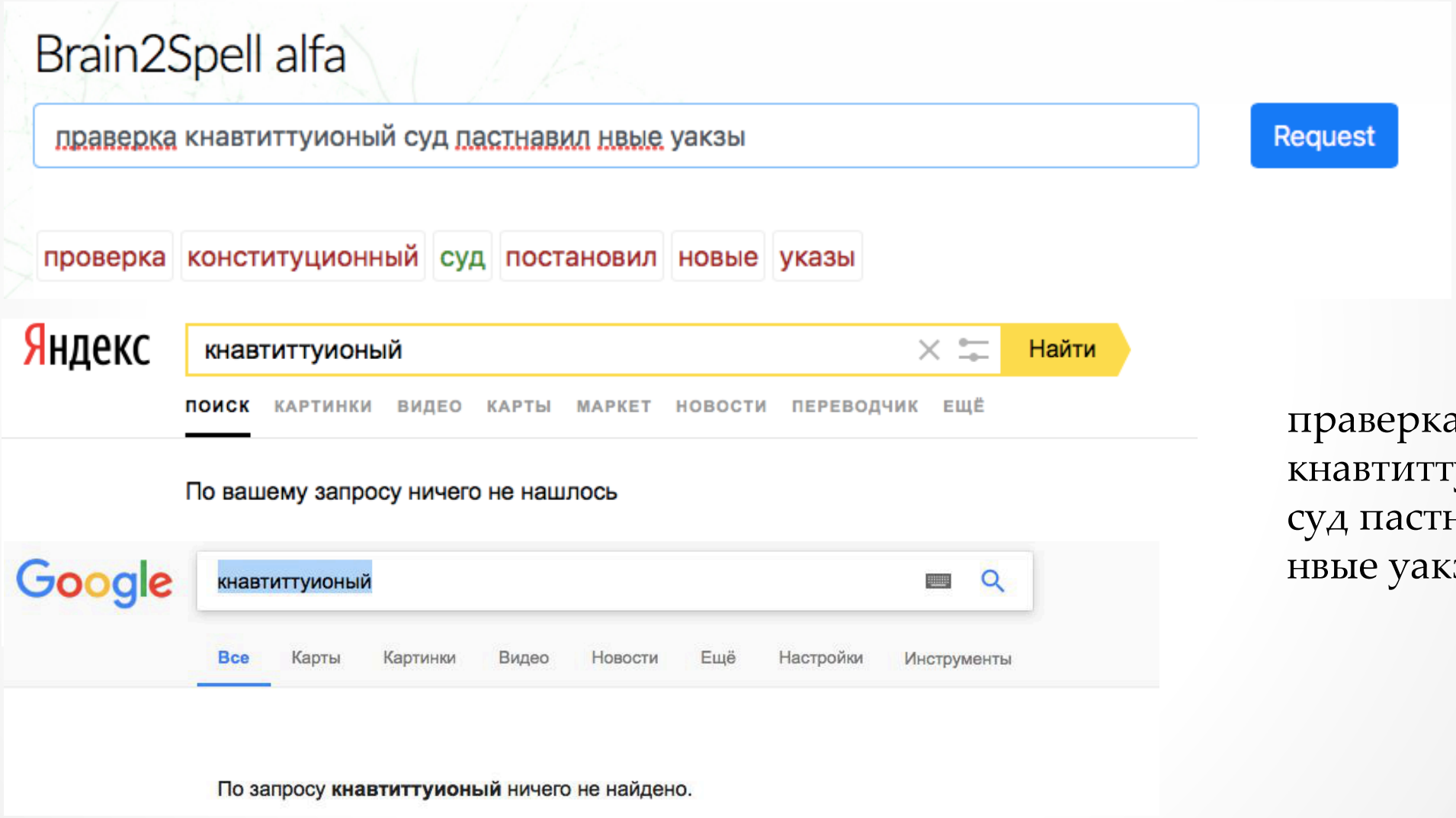
Spelling Error Models

Sources:

1. Correcting Writing Errors with Convolutional Neural Networks
   https://scholar.colorado.edu/cgi/viewcontent.cgi?referer=https://www.google.ru/&httpsredir=1&article=1128&context=csci_gradetds
2. Correcting Grammatical Errors
   https://ufal.mff.cuni.cz/pbml/108/art-tezcan-hoste-macken.pdf
3. A Comparison of Standard Spell Checking Algorithms and a Novel Binary Neural Approach
   http://eprints.whiterose.ac.uk/689/1/hodgevj2.pdf
4. Spell Ru Eval: the first competition on automatic spelling correction for Russian
   https://www.academia.edu/25881770/SpellRueval_the_First_Competition_on_automatic_Spelling_Correction_for_Russian?auto=download
5. Learning a Spelling Error Model from Search Query Logs
   https://dl.acm.org/citation.cfm?id=1220695
6. Автоматическое исправление опечаток в поисковых запросах без учета контекста –
   http://www.dialog-21.ru/media/1276/paninamf.pdf
7. A Complex Approach to Spellchecking and Autocorrection for Russian
   http://www.dialog-21.ru/media/3450/derezaovetal.pdf
8. Fast and Accurate Misspelling Correction in Large Corpora
   http://www.dialog-21.ru/digests/dialog2015/materials/pdf/ShavrinaTOSorokinAA.pdf
9. Моделирование расширенной лемматизации для русского языка на основе морф. парсера TnT-Russian
   http://repository.tudelft.nl/islandora/object/uuid:dd1a7705-9662-4e2c-ab33-e8d228993dac/datastream/OBJ

Spelling Error Models

http://cogsys.company/en/brain2spell



праверка кнавтиттуионый суд пастнавил нвые уакзы

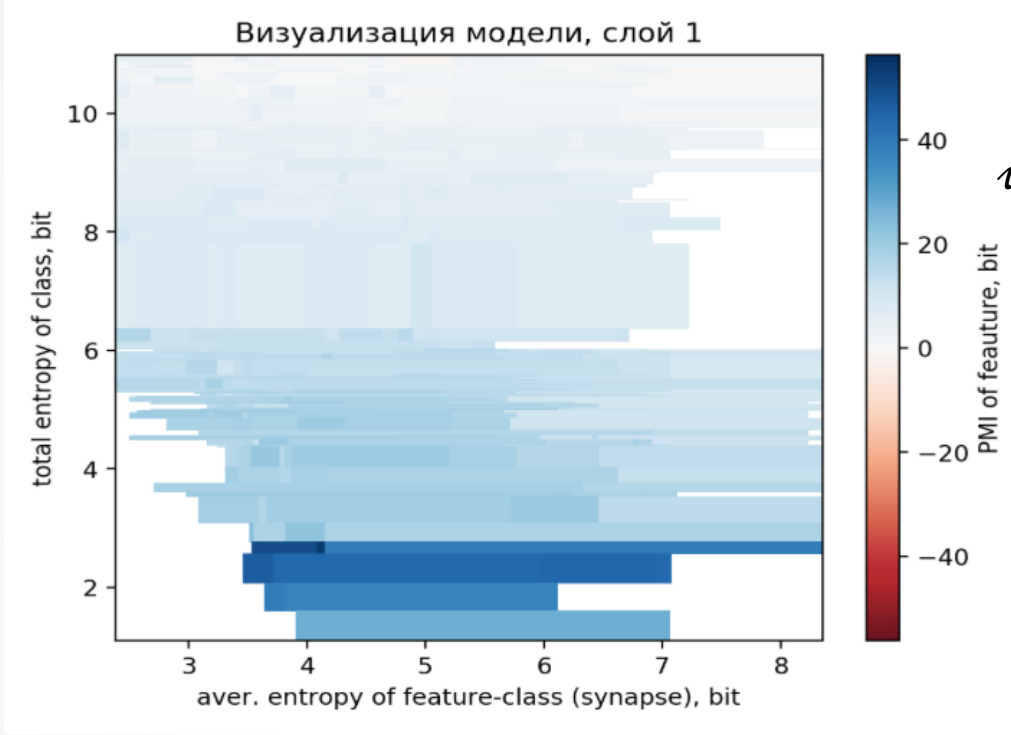
Spelling Error Models

CORRECT	MISTAKE	MS_WORD	Brain2Spell	MS_WORD	Brain2Spell
мужественный	мужиствиний	мужественней	мужественный	0	1
аплодисменты	обладесменты	обладесменты	аплодисменты	0	1
экскурсии	екскурся	экскурса	экскурсии	0	1
пятьдесят	педясят	подаст	пятьдесят	0	1
путешественник	путишествинек	путешественник	путешественник	1	1
препинания	припеннание	препинание	препинания	0	1
пассажирский	посожирсский	пассажирский	пассажирский	1	1
человек	ччеловввек	человеке	человек	0	1
большая	большаааая	большущая	большая	0	1
приспособление	присобление	присоленные	приспособление	0	1
шестьдесят	шесдисят	шестисот	шестьдесят	0	1
искусство	иссскуство	искусство	искусство	1	1
рассказывать	разказивать	размазывать	рассказывать	0	1
рассказывать	розказывать	размазывать	рассказывать	0	1
ассоциации	асоцыацыи	ассоциации	ассоциации	1	1
профессионал	проффесианал	профессионал	профессионал	1	1
территория	тереторрия	территория	территория	1	1
аплодисменты	опладесменты	аплодисменты	аплодисменты	1	1
ремесленное	ремеслянная	ремесленная	ремесленное	0	1
опасность	опасносность	опасносность	опасность	0	1
карандаш	карандандаш	карандандаш	карандаш	0	1
эволюционировать	эволюцировать	эволюцировать	эволюционировать	0	1
лаборатория	лаборатораприя	лаборатораприя	лаборатория	0	1
соревноваться	соревновноваться	соревновноваться	соревноваться	0	1
оптимист	оптимисомст	оптимисомст	оптимист	0	1
классный	классняшный	краснушный	классный	0	1
похитить	похититттть	похититттть	похитить	0	1
фотоаппарат	фотопорат	фотопират	фотоаппарат	0	1
фотоаппарат	фотопарад	фото парад	фотоаппарат	0	1

http://cogsys.company/en/brain2spell

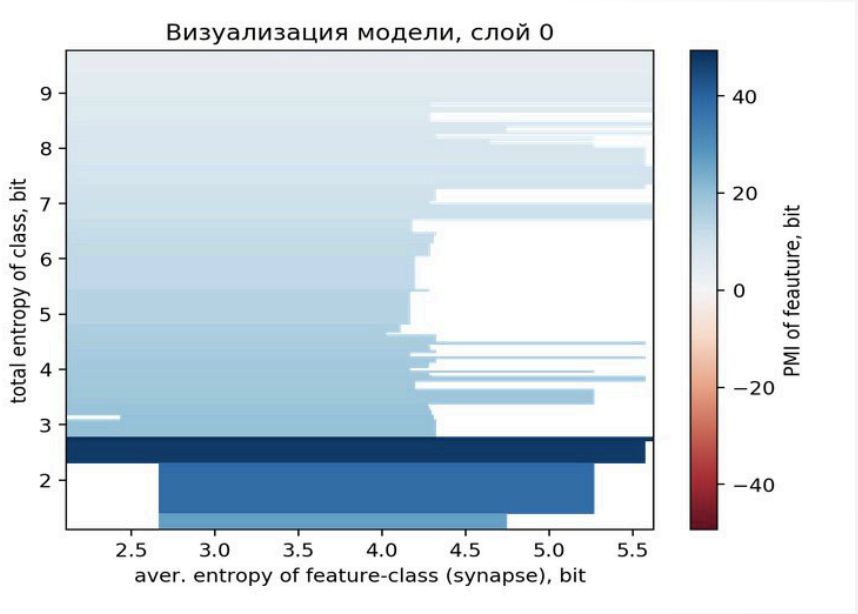
Example of entropy distribution

h(w)



h(z)

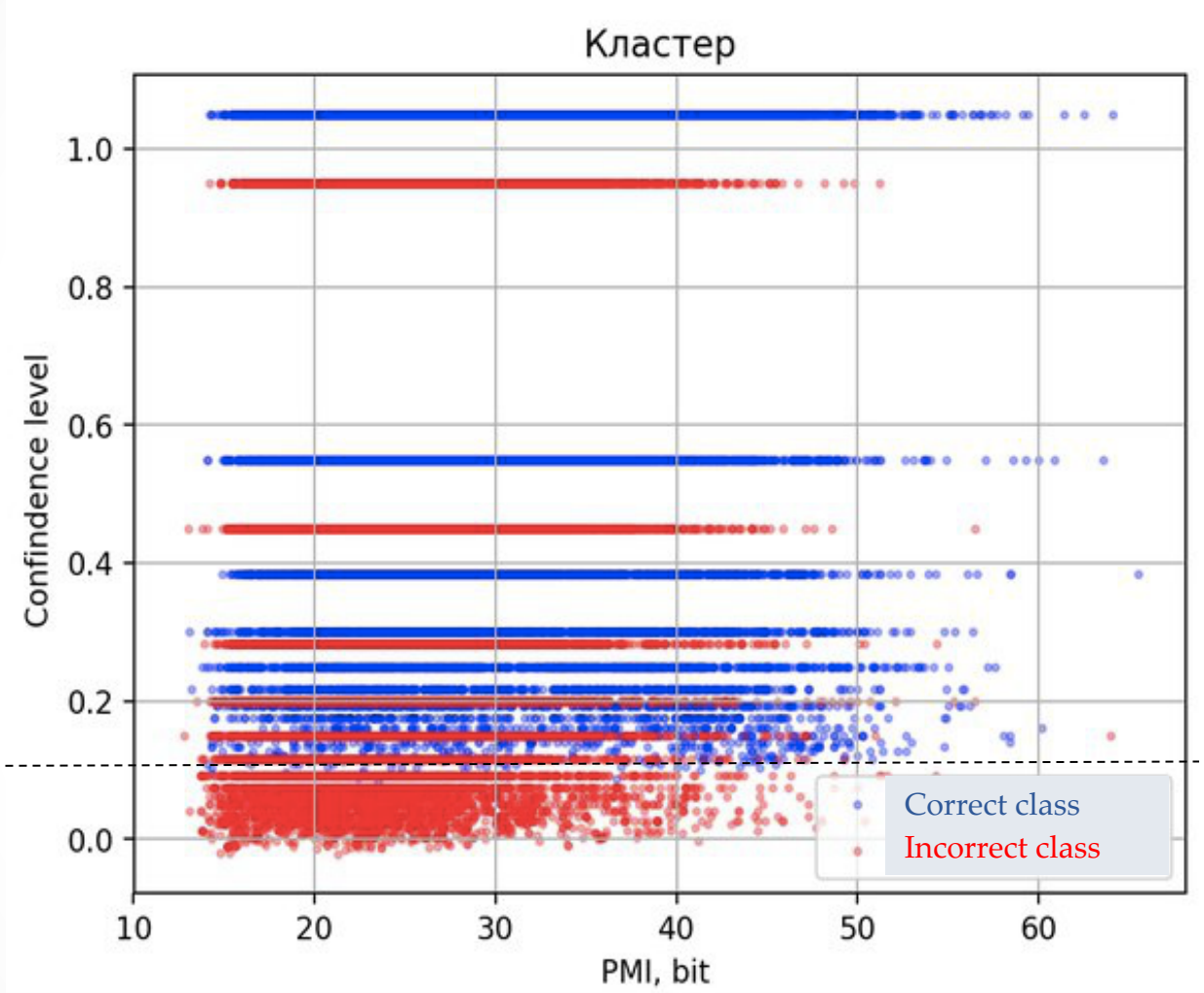
$wNPMI(w,m)$



Distribution of entropy for the 2-nd layer of lexicon model (letters in the center of the word)

Distribution of entropy for the 1-st layer of lexicon model (first letter of the word)

Example of confidence level idea



Critical level for classification (= any answer is very creative)

Distribution of correct and incorrect objects' classification

Brain2 Framework – INA Application

Фильтр классов модели по микро F1-мере

от 0 до 1

Установить пороговые значения

F1-мера (макро):	0.387
Точность:	0.375
Полнота:	0.400
Энтропия:	0.666

Параметров:	140
Классов:	2
Интервалов классов:	14
Признаков:	3
Интервалов признаков:	10
Примеров для обучения:	7
Примеров для тестирования:	3
Ненулевых связей:	44

Model information: quality indicators

Confusion matrix (heat map)

Фильтр классов модели по микро F1-мере

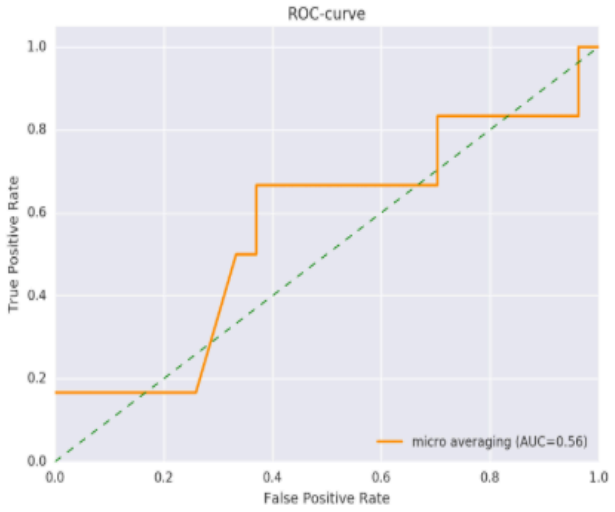
от 0 до 1

Установить пороговые значения

F1-мера (макро):	0.387
Точность:	0.375
Полнота:	0.400
Энтропия:	0.666

Параметров:	140
Классов:	2
Интервалов классов:	14
Признаков:	3
Интервалов признаков:	10
Примеров для обучения:	7
Примеров для тестирования:	3
Ненулевых связей:	44

ROC-кривая 20 скачать csv



cena

0: 1000.0:1500.0 **1:** 1750.0:1875.0 **2:** 5000.0:5500.0

3: 5500.0:6000.0

level

4: high **5:** low **6:** middle

Model information: quality indicators

ROC - Curve

Фильтр классов модели по микро F1-мере

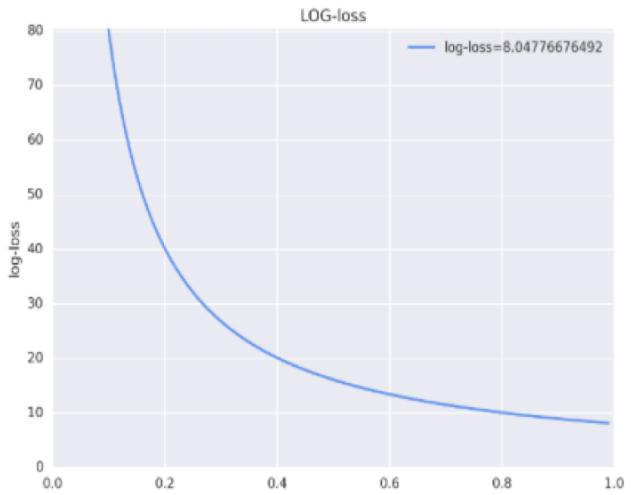
от 0 до 1

Установить пороговые значения

F1-мера (макро):	0.387
Точность:	0.375
Полнота:	0.400
Энтропия:	0.666

Параметров:	140
Классов:	2
Интервалов классов:	14
Признаков:	3
Интервалов признаков:	10
Примеров для обучения:	7
Примеров для тестирования:	3
Ненулевых связей:	44

Логистическая кривая 20 скачать csv



cena

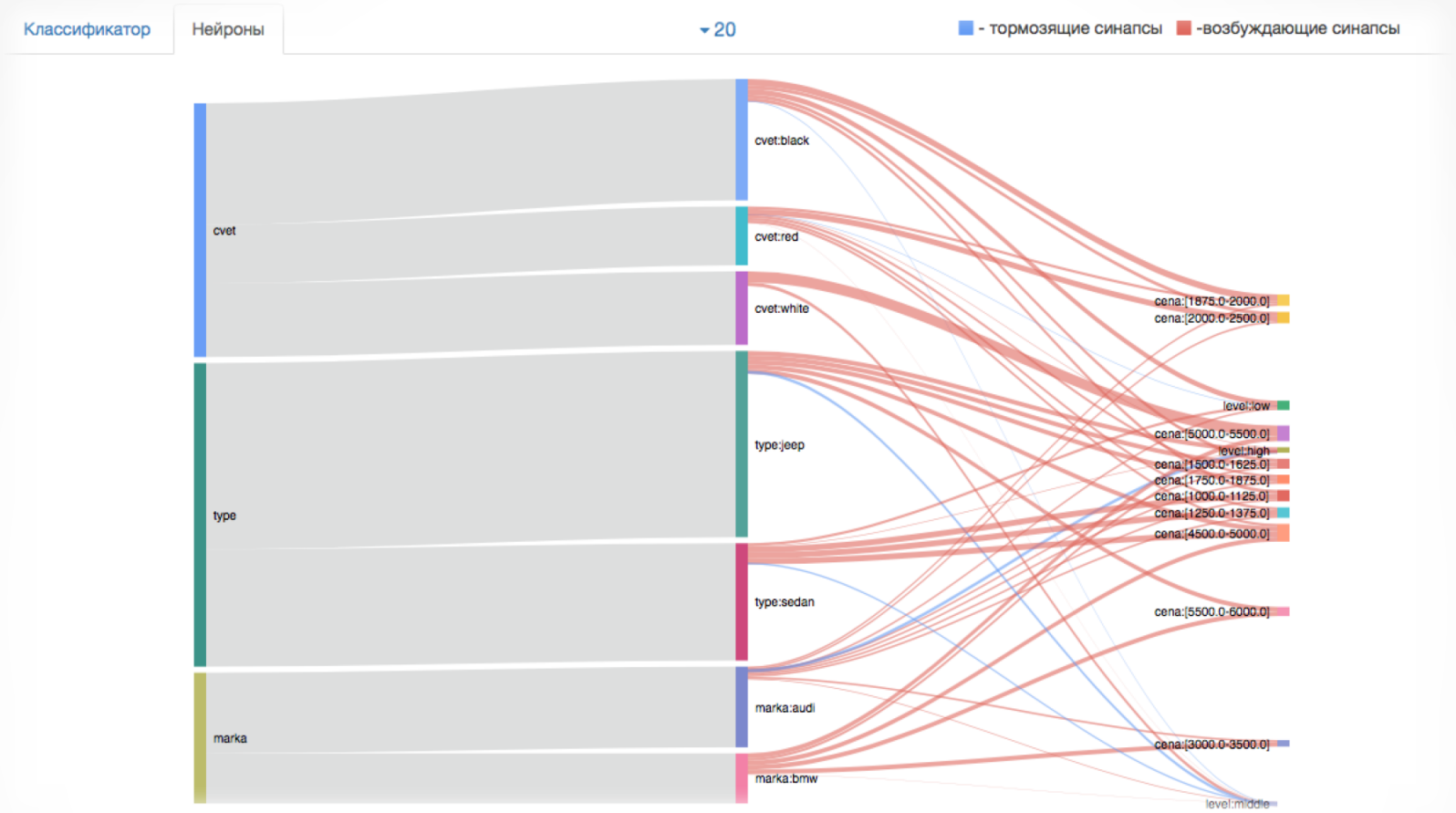
0: 1000.0:1500.0 **1:** 1750.0:1875.0 **2:** 5000.0:5500.0 **3:** 5500.0:6000.0

level

4: high **5:** low **6:** middle

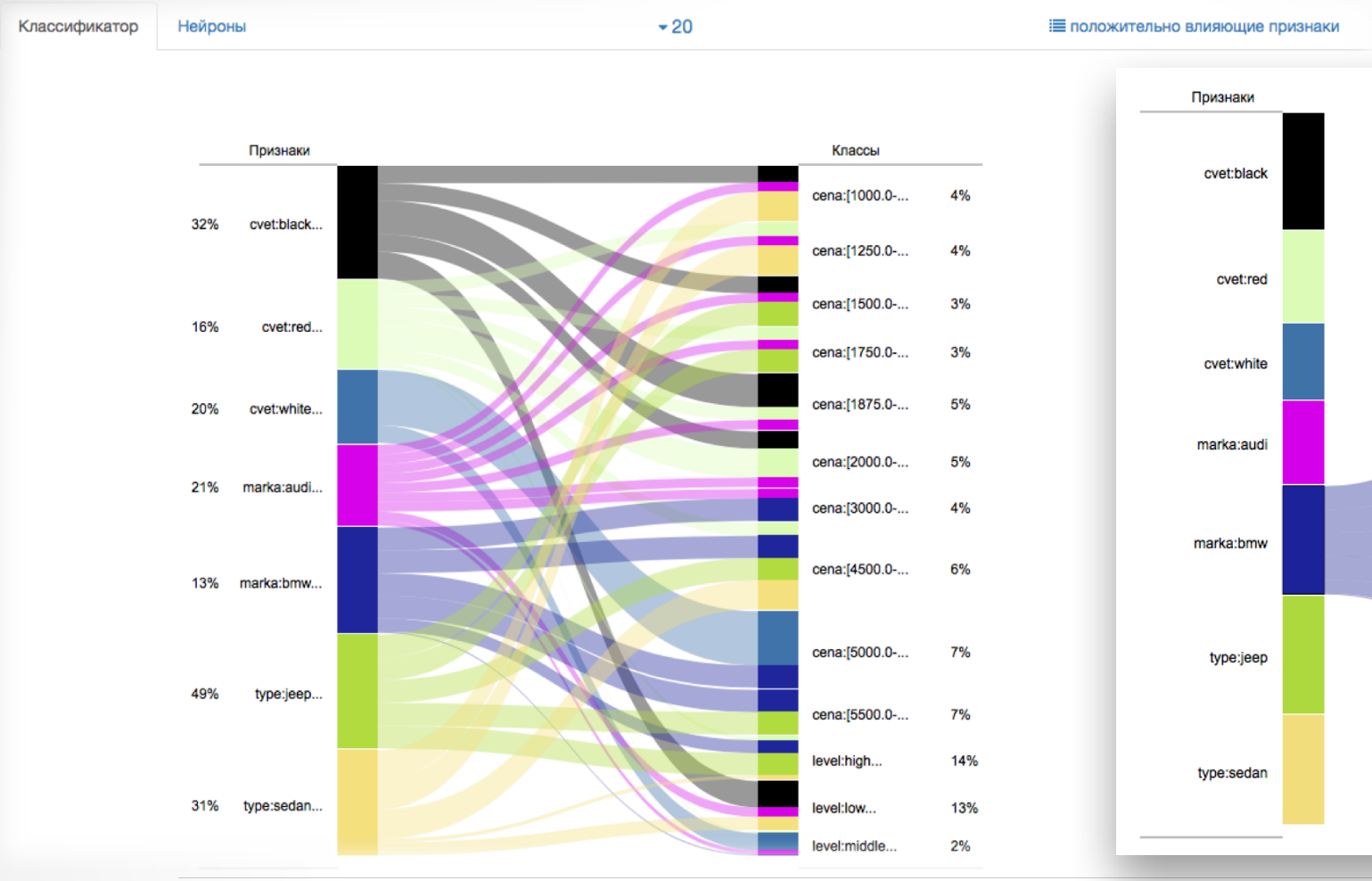
Model information: quality indicators

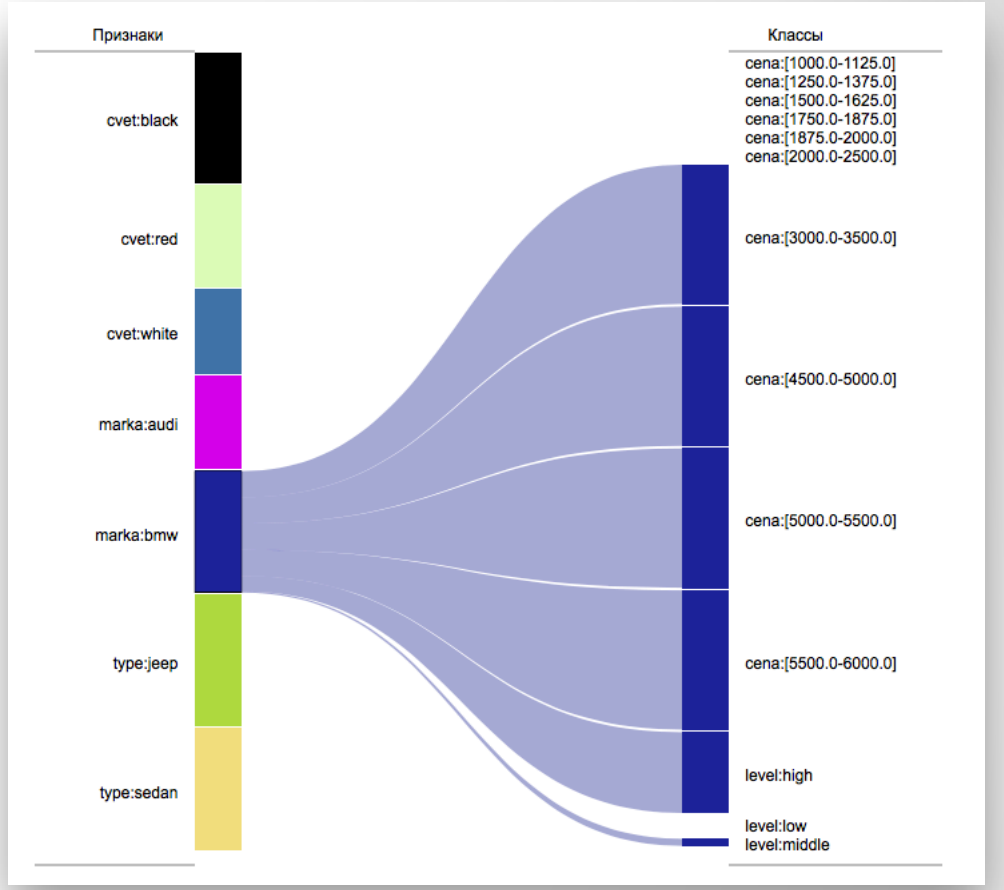
Logistic Curve



Model information: model visualization

The weights of the links between class and feature intervals are visualized, as well as the informational significance of the feature intervals. It is possible to plot a chart for the selected intervals of classes and/or features with a certain informational significance.





Model information: model visualization

The weights of the links between class and feature intervals are visualized, as well as the informational significance of the feature intervals. It is possible to plot a chart for the selected intervals of classes and/or features with a certain informational significance.

Framework Technical Specifications

- **Python 3.6**
  - **numpy**
    - array, vectors
    - vectorize (elementwise functions)
  - **pandas** (dataset reading, statistics)
  - **scipy**
    - sparse matrices
    - mathematical functions
  - **nltk** (lemmatizator, tagging)
  - **sklearn.feature_extraction** (text vectorizing)
  - **sklearn.metrics** (metrics)
  - **multiprocessing** (process paralleling)
  - **Flask**, **Tornado** (web, async)

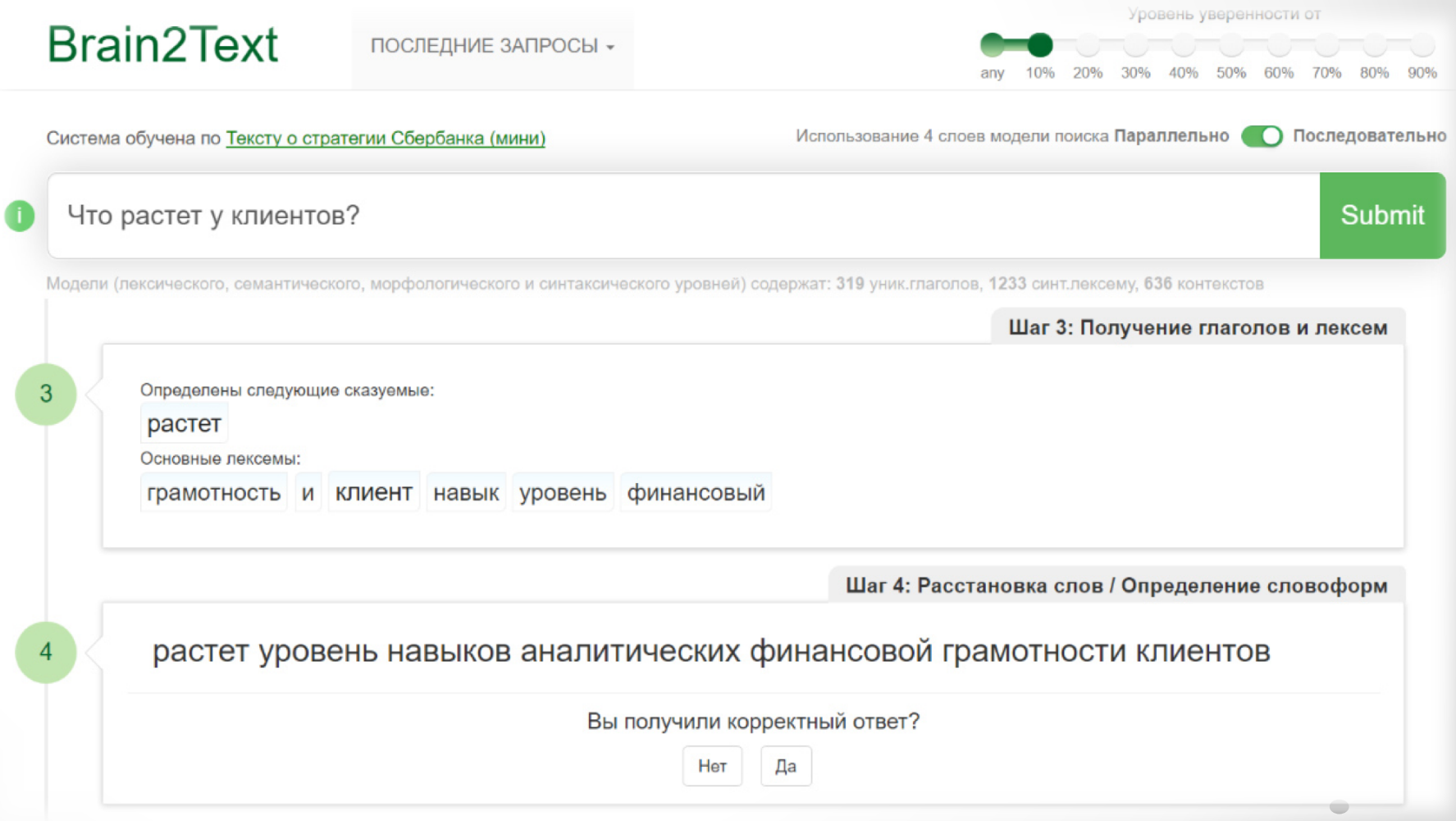
- **Operating System**
  - Centos 6.9 64 Bit
- **Hardware**
  - Intel® Xeon®, E5-1650 v3 Hexa-Core Haswell – 6 core
  - 128 GB ECC RAM
  - 2x 240 GB 6 Gb/s SATA SSD Data Center Series

Admin Home Page

Information about the load on the server. You can see both statistical (the top part of the graph) and dynamic information.

Brain2Text Case

- Trained on natural text
- 5 multilayer models
- Understands the question and conjures up the answer
- No pre-set questions and answers – the number of questions is unlimited



http://sb.brain2.online/

The Way to Connect Discrete and Continuous Quantities

		Humidity	*Mean*	*StDev*
Play Golf	yes	86 96 80 65 70 80 70 90 75	79.1	10.2
	no	85 90 70 95 91	86.2	9.7

$$P(\text{humidity} = 74 \mid \text{play} = \text{yes}) = \frac{1}{\sqrt{2\pi}(10.2)} e^{-\frac{(74-79.1)^2}{2(10.2)^2}} = 0.0344$$

$$P(\text{humidity} = 74 \mid \text{play} = \text{no}) = \frac{1}{\sqrt{2\pi}(9.7)} e^{-\frac{(74-86.2)^2}{2(9.7)^2}} = 0.0187$$

Thank you for attention!

More information at cogsys.company