PAQ compression

Krzysztof Blaszczyk

Overview

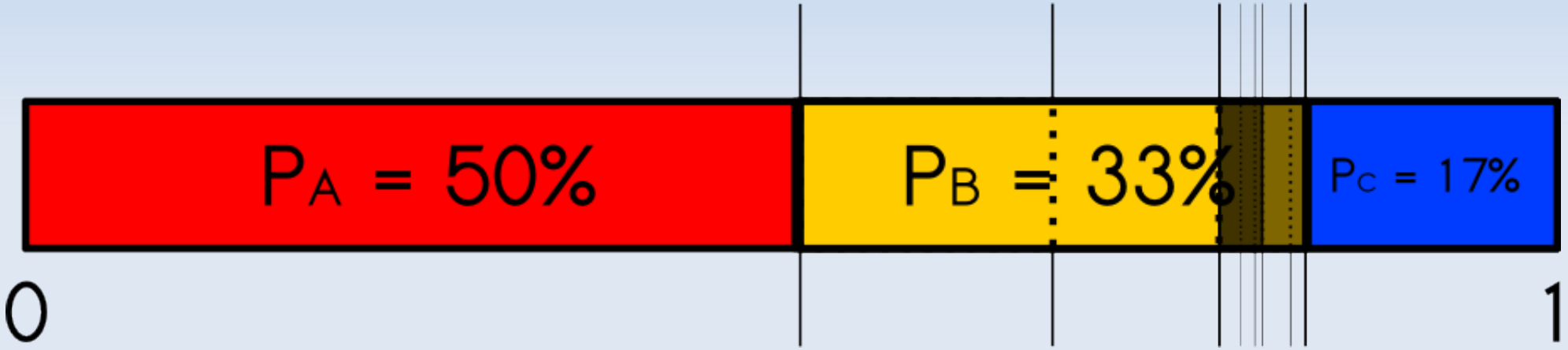
- Introduction: What is PAQ?
- Aritmetic coding and PPM recap
- PAQ models and contexts
- Context mixing
  - PAQ 0-3.x
  - PAQ 4-6.x
  - PAQ 7+ and Neural Networks
- Optimazations
- Improvement suggestions
- Advantages/Disadvantages
- Comparisions

What is PAQ?

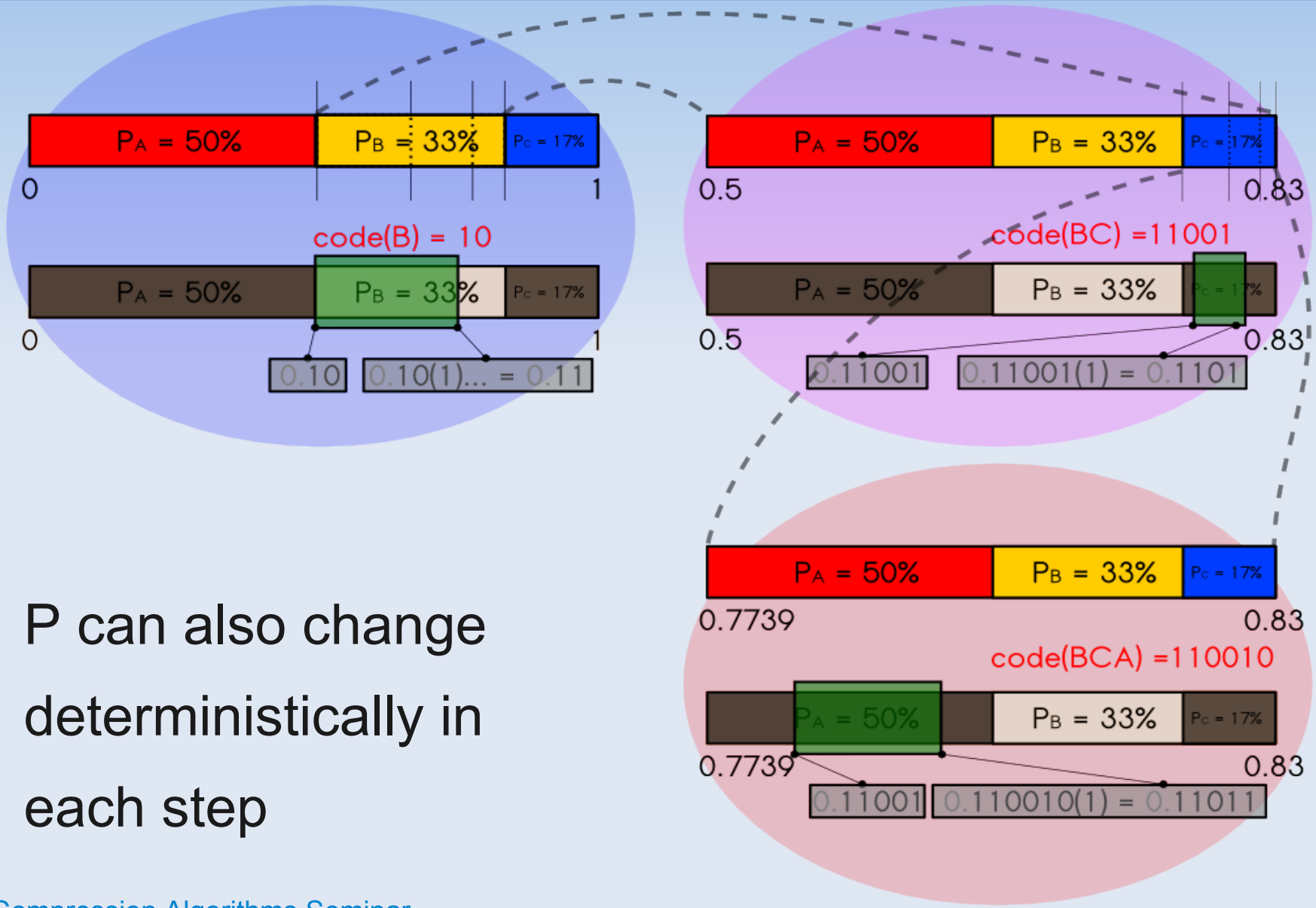
- Series of opensource file archivers
- Advanced compression algorithm
- First PAQ version was released 2002
- Since then evolved through competition, experimenting, trial and error
- Slow, but pushes the compression to its theoretical limits
- ZPAQ 4.04 introduces backwards compatibility
- Commercial competiton is WinRK

Recap: Arithmetic coding

- Recursively subdivide interval [0,1)...

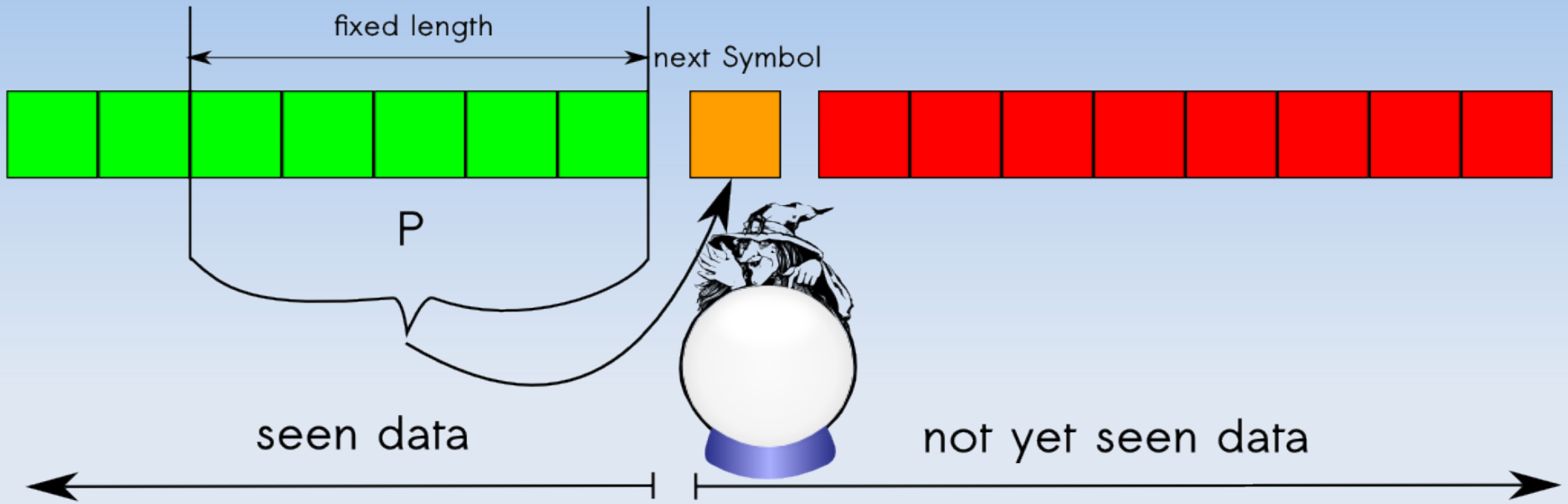


Recap: Arithmetic coding



- P can also change deterministically in each step

Recap: PPM



- Data window is some recent part of already seen data
- Prediction is done only once per symbol by assuming that some data pattern occuring in the window will continue occuring

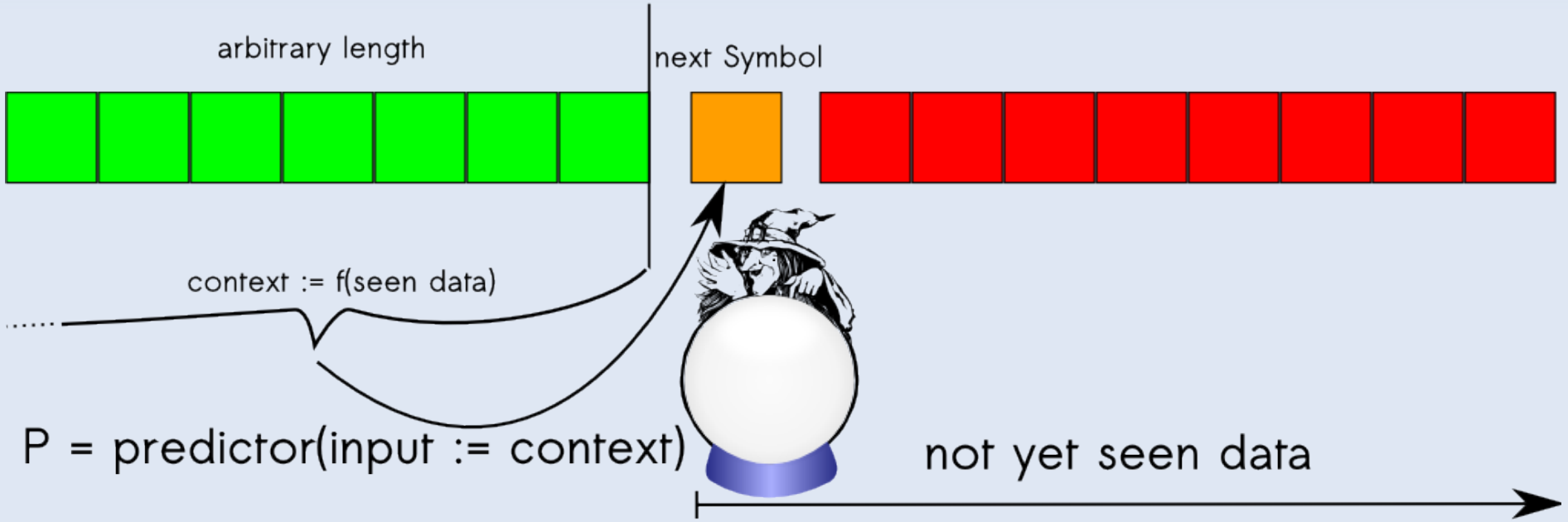
PAQ Introduction

- PAQ is an extended version of PPM that uses arithmetic coding with

  $\Sigma = \{0,1\}$ and $P = (P(0),P(1))$
- P is adaptive
- Depending on the PAQ version, (P(0),P(1)) can be expressed as a pair of integer counts or a pair of doubles

How does PAQ work?

- Contexts are now arbitrarily complex functions of ”already seen” data
- They define the input that a predictor receives

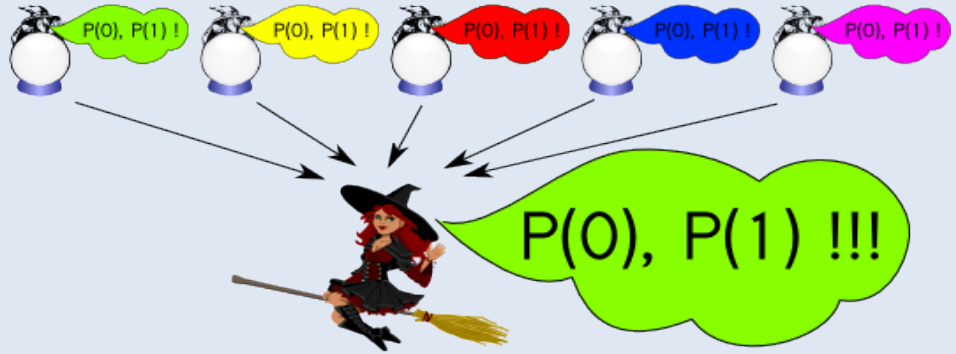


How does PAQ work?

- More importantly PAQ generalizes the prediction process into an array of multiple models which are mixed into one single model
- Therefore the final prediction is based on more world knowledge and tends to be more accurate
- Prediction process must remain deterministic for decompression

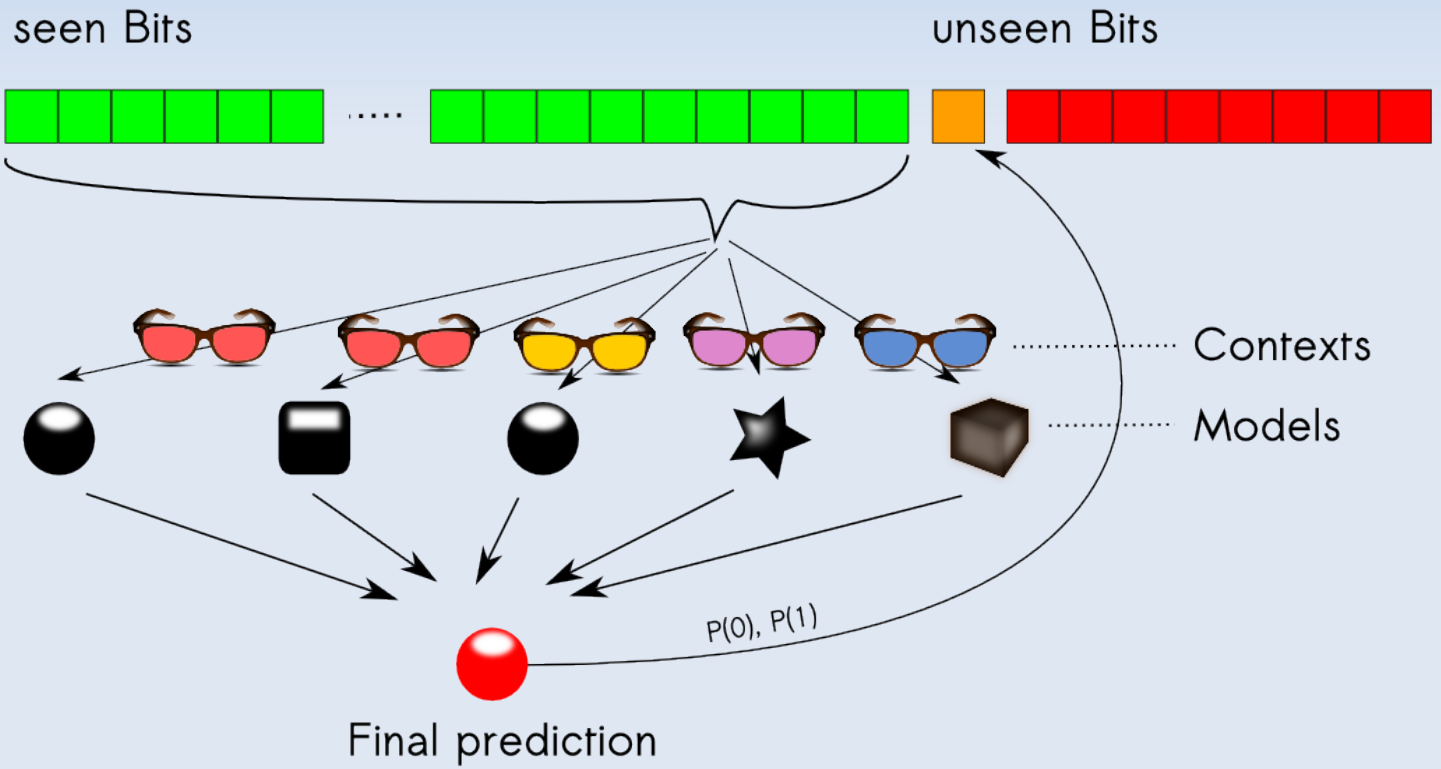
BEFORE:

AFTER:



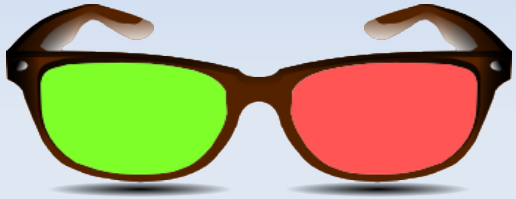
Models

- In PAQ, a model is a prediction function that attempts to predict ***one*** single bit at a time. The input is some binary sequence and the output expresses the probability distribution for the next following bit.
- Each model must be given so called context in order to make a prediction



Contexts

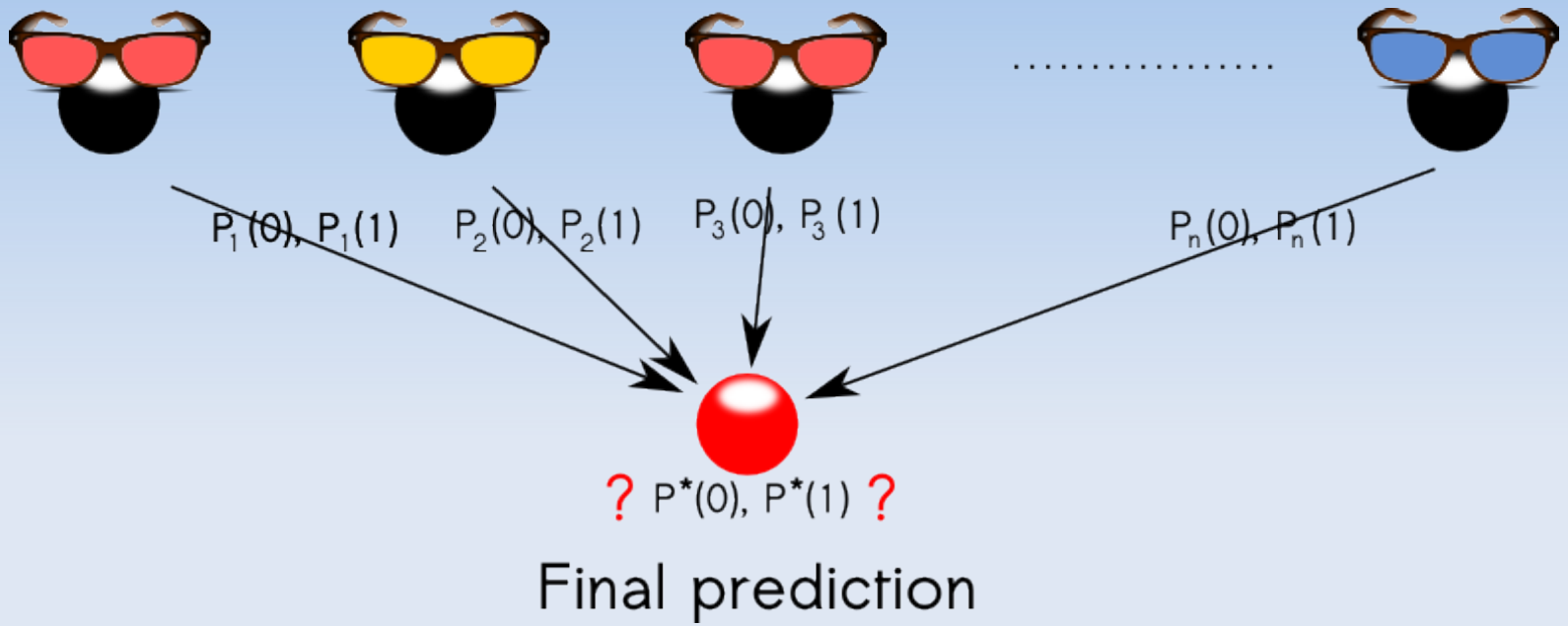
- A context is a function of previously seen data
- The output of a context defines the input for a model
- Examples of contexts:
  - N-gram: last raw n bytes before the predicted symbol
  - A fixed string
  - The hash value of the last 20 bits
  - Selection of high order bits from an N-gram



Model example

- Model receives some sequence Q of 16 bits that was defined by associated context
- The model contains some assumption about the input, for example that there should be 30% ones and 70% zeroes
- If Q consists of 50% ones, than the model detects a lack of zeroes under this assumption and expresses a higher probability that a zero will follow
- Another example would be a model ”wave” that might assume that the data values follow some sinus-like pattern and perform prediction by approximating the data with some sinus function.

Context mixing



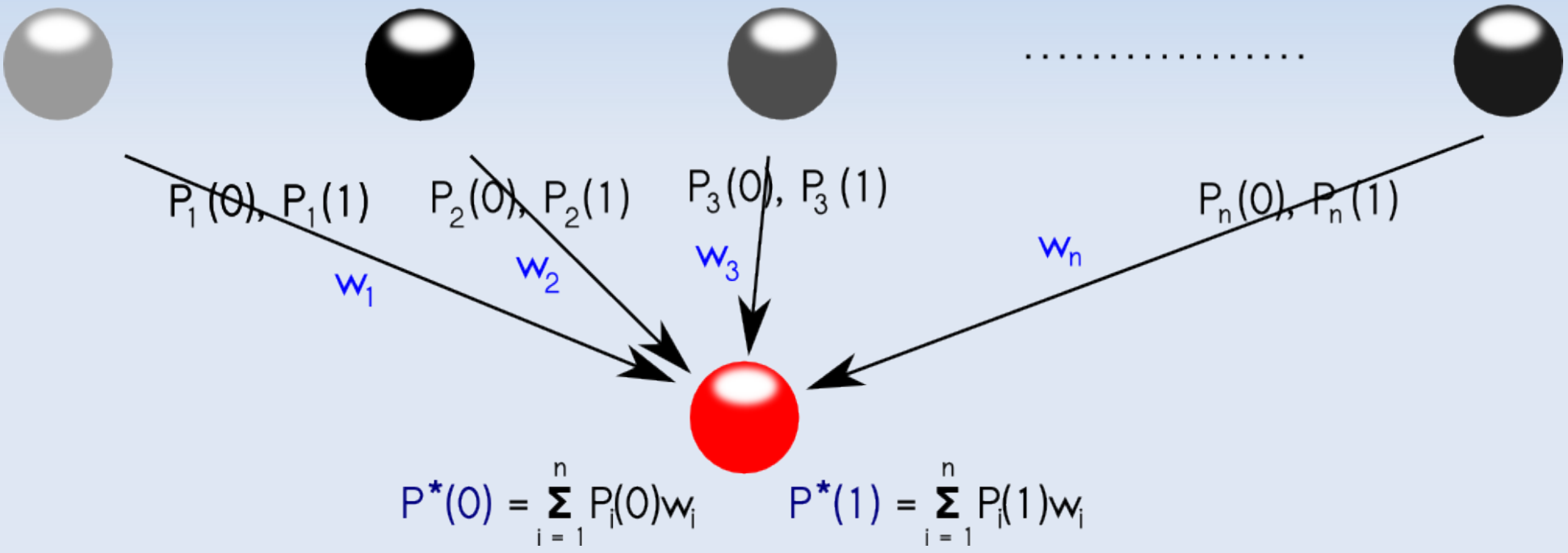
- How to combine the multiple probabilities ($P_i(0)$,$P_i(1)$) estimated by different models into one single probability distribution?

Mixing by model weighting in PAQ 0-3.x

- Assign fixed weights to each model W = {w1,...,wn}
- The greater the context the greater the weight of the model
- Each model expresses the probability by maintaining a pair of integer counts $P_i(0)$ and $P_i(1)$
- For example if $(P_i(0), P_i(1)) = (7,4)$, then $P(0) = 7/(4+7) \sim 0.64$ and $P(1) = 4/(4+7) \sim 0.36$
- Let $(P^*(0), P^*(1))$ be the final combination of all predictions $(P_i(0), P_i(1))$
- To calculate $(P^*(0), P^*(1))$ weighted averaging is performed

Mixing by model weighting in PAQ 0-3.x

- To calculate (P*(0), P*(1)) weighted averaging is performed

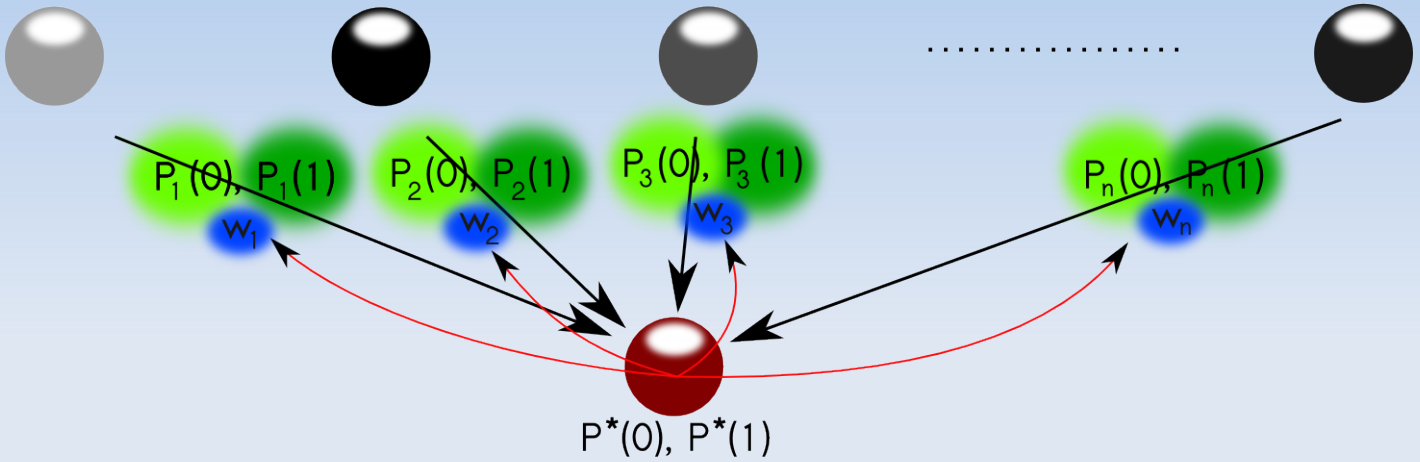


Mixing by adaptive model weighting in PAQ 4-6.x

- Problem to solve: Some local data patterns might have been predicted better with different model weights
- Solution: adjusting the weights dynamically opens the possibility of adaptation to local patterns.

Mixing by adaptive model weighting in PAQ 4-6.x

Let x be the bit to be coded



$$Sum0 = P^*(0) = \sum_{i=1}^{n} P_i(0) w_i$$

$$Sum1 = P^*(1) = \sum_{i=1}^{n} P_i(1) w_i$$

$$Sum = Sum0 + Sum1$$

$$\frac{Sum0}{Sum1} = \frac{P_i(0)}{P_i(1)}$$

$$error = x - \frac{Sum1}{Sum}$$

$$w_i = w_i + error \quad \frac{Sum0 \cdot P_i(1) - Sum1 \cdot P_i(0)}{Sum0 \cdot Sum1}$$

Mixing by adaptive model weighting in PAQ 4-6.x

- Adjust counts for each model that was wrong.

  Let x be the actual symbol that model i tried to predict.

  - If 0 was predicted ($P_i(0) > P_i(1)$) but x=1 then

    $(P_i(0), P_i(1)) := ((P_i(0)-2)/2, P_i(1)+1)$ if $P_i(0) > 2$

  - If 1 was predicted ($P_i(0) < P_i(1)$) but x=0 then

    $(P_i(0), P_i(1)) := (P_i(0)+1, (P_i(1)-2)/2)$ if $P_i(1) > 2$

Neural Network mixing in PAQ 7+

- Previous fromula suggests to apply the idea of neural networks to adjust weights

$$w_i = w_i + \text{error} \cdot \frac{\text{Sum0} \cdot P_i(1) - \text{Sum1} \cdot P_i(0)}{\text{Sum0} \cdot \text{Sum1}}$$

$$w_i = w_i + \text{error} \cdot \alpha \cdot \text{Input}_i$$

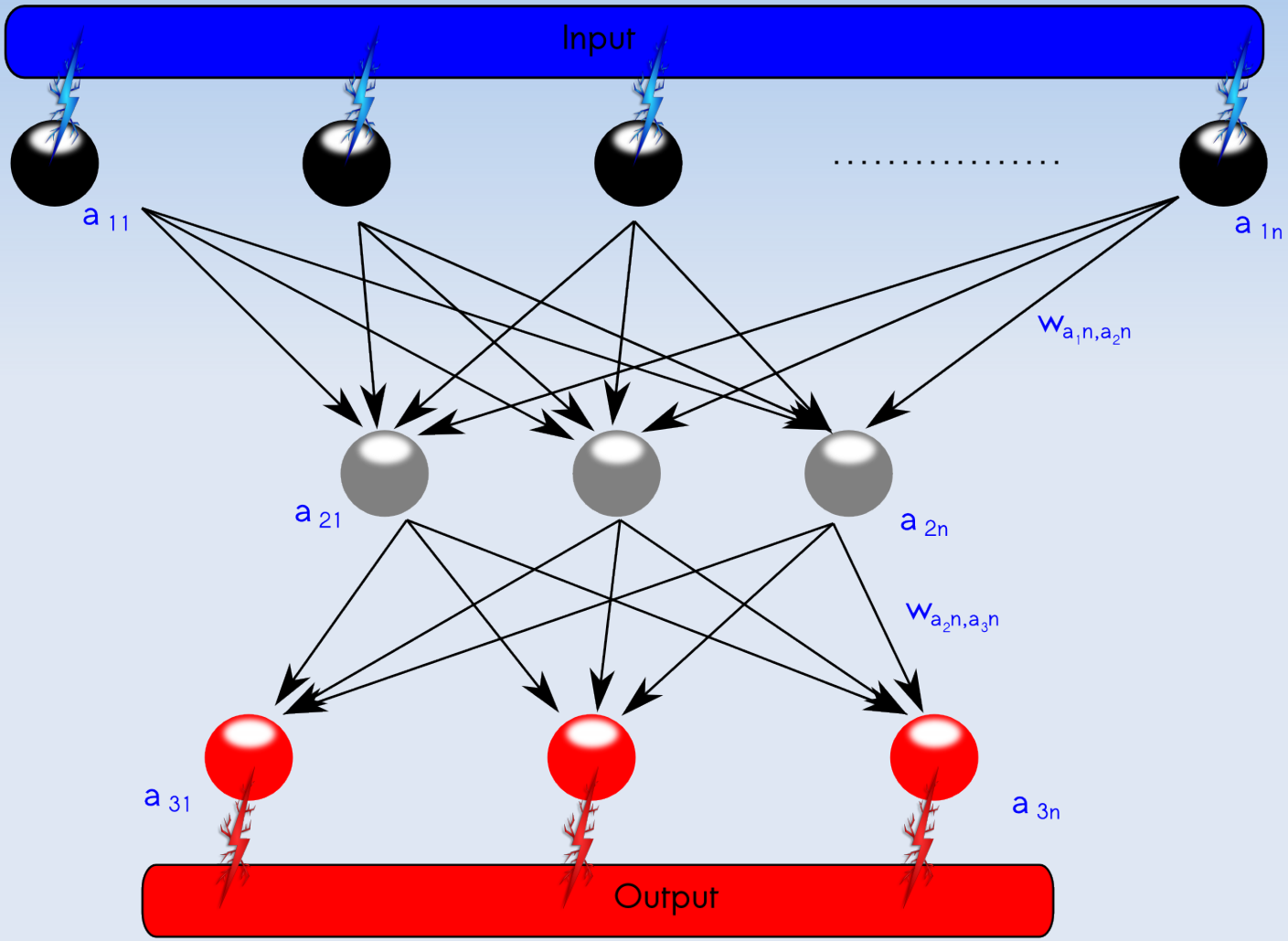
learning rate

$$f\left(\sum_{i=0}^{n} \text{in\_signal}_i\right)$$

Neural Network mixing in PAQ 7+

- Hutter (organizer of the Hutter prize) claims in his book on universal AI that ”the optimal behavior of a goal seeking agent in an unknown but computable environment is to guess at each step that the environment is probably controlled by one of the shortest programs consistent with all interaction so far” (Source: Wikipedia)
- In his view compression is an AI problem
- So what is a neural network?

Neural Networks



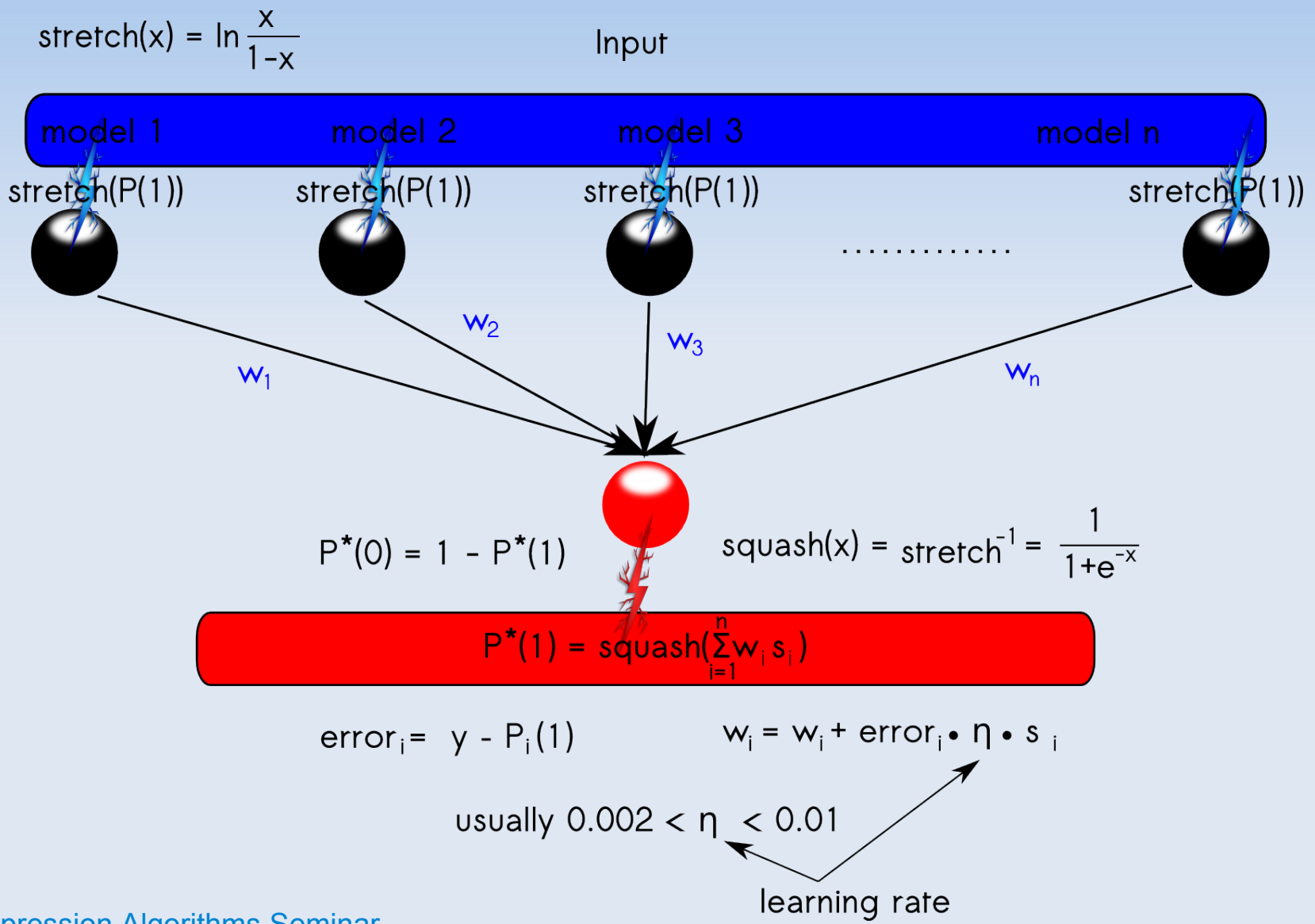
Backpropagation Neural Network

- The input nodes fire signals of some strength
- The signal gets distributed over all edges and multiplied by the weight of the edge that it passes
- The signal strength of the output nodes is a function of the sum of all incoming signals
- The interpretation what a strong or weak signal means can be chosen freely
- Backpropagation networks are used if the ideal output is known. If so then we can calculate an error for each output node and adjust the weights by going backwards from Output to Input

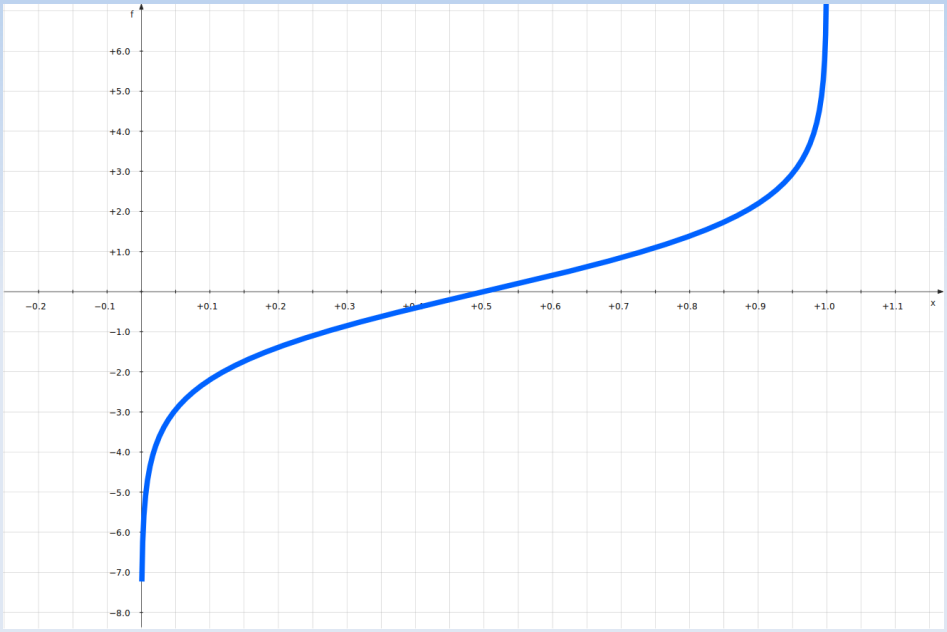
Backpropagation NN in PAQ 7+

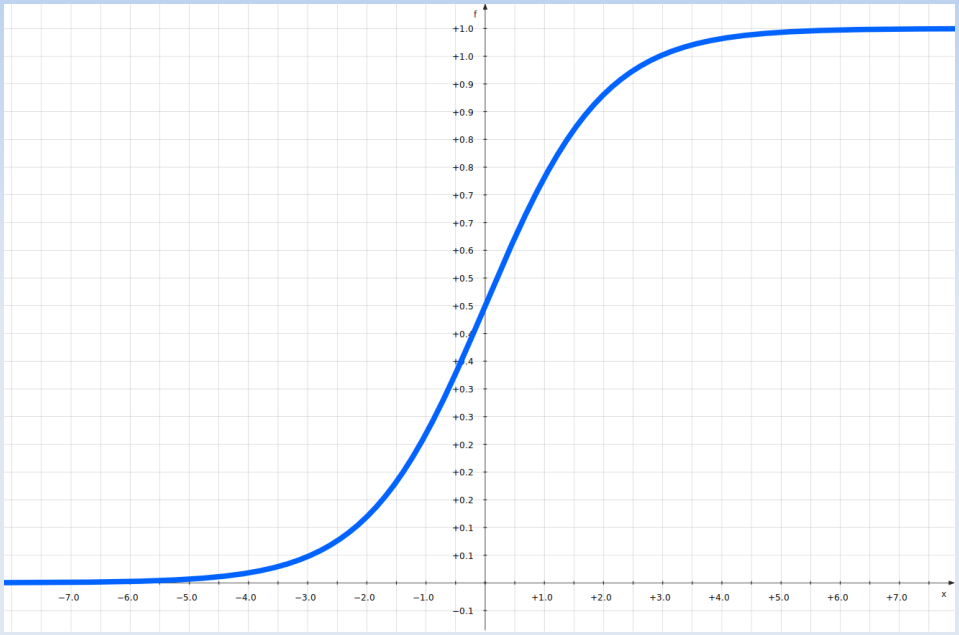
- $P_i(0)$, $P_i(1)$ are rational numbers in [0,1] now
- Each input neuron represents the prediction of one distinct model. The signal s that is fired by each input node is a function of $P_i(1)$
- Output layer consists of only one single node that represents the total proability P*(1) (Obviously P*(0)= 1 - P*(1)

Neural Network mixing in PAQ 7+



Logit function





stretch(x) = logit(x)

squash(x) = $\text{logit}^{-1}(x)$

Optimazations

- PAQ recognizes file formats and can choose specialized models/contexts that target only a specific format i.e JPEG's
- newer PAQ versions preprocess the data before compressing it (i.e in texts by using a dictionary to replace words with their dictionary indexes)

Improvement Suggestions

- Skip ”unpredictable” data chunks
- Deterministically evolve data specific models i.e by making use of more AI algorithms

Disadvantages/Advantages

- Disadvantages
  - Slow and memory consuming
  - Compression/Decompression takes the same amount of time and memory
- Advantages
  - Developers believe, that PAQ is not encumbered by any patents
  - Free/Open source
  - Best compression ratios

Comparision with other Archivers

	TEXT	BMP	ZIP	JPEG	MP3	RANDOM
PAQ8O	19.95%	30.86%	96.31%	83.15%	94.13%	100.060%
7ZIP	28.04%	68.29%	98.38%	100.1%	98.23%	100.007%
ZIP	34.6%	76.68%	98.61%	99.89%	98.61%	100.001%
RAR	30.66%	37.93%	98.69%	100.2%	98.18%	100.025%

More at: http://www.maximumcompression.com/index.html