Recommendation system for scientific tools and workflows

Master's thesis

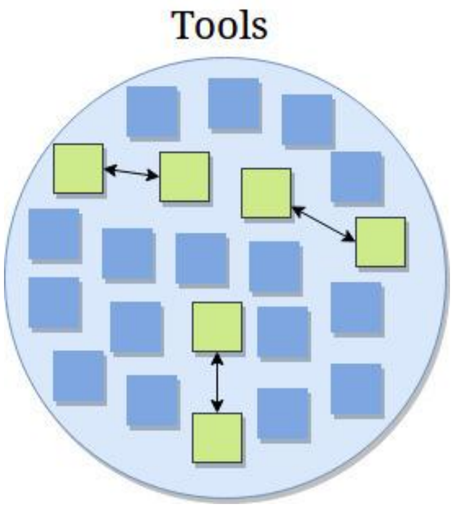
07/08/2018

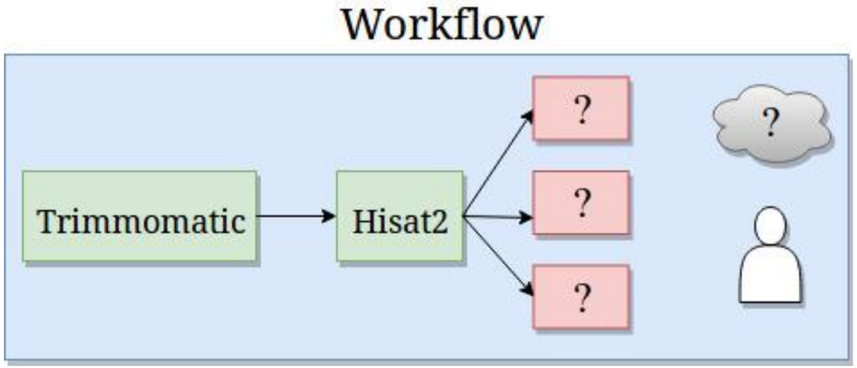
Anup Kumar

Adviser: Dr. Björn Grüning

Motivation

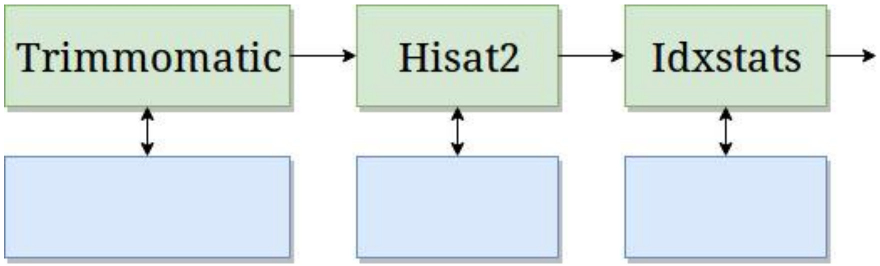
- Galaxy - biological data analysis
- Tools and workflows
- Large number of tools (> 1,000)
- Complex workflows
- Need guidance - recommendation system



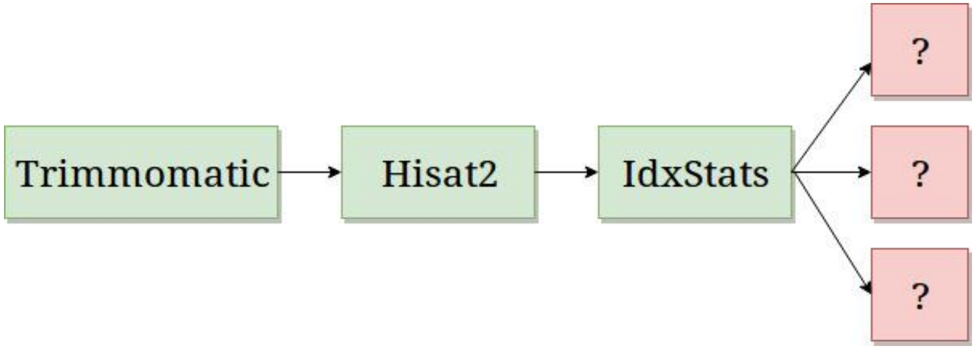


Recommendation system

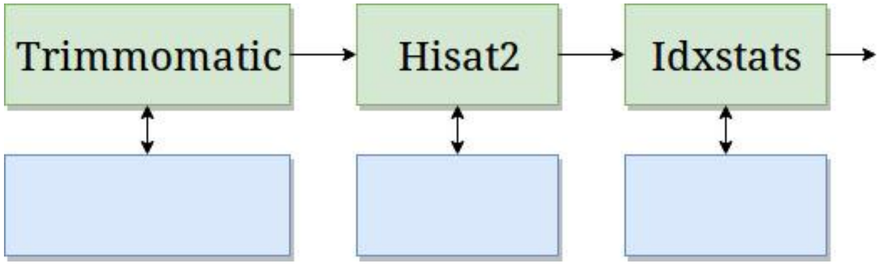
1. Find similar scientific tools



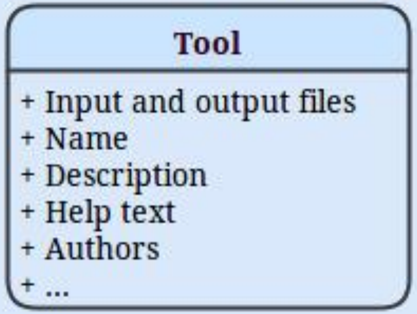
2. Predict tools in workflows



1. Find similar scientific tools

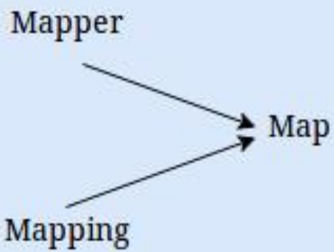


1. Read tool XML files

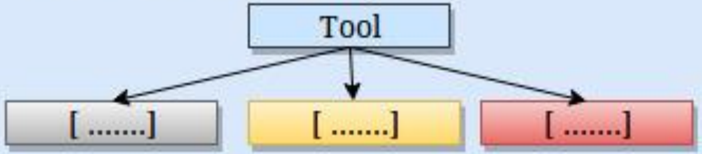


2. Clean metadata

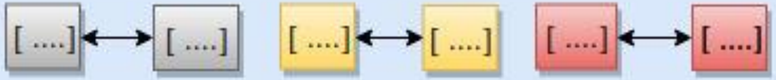
- Stopwords: "at, in, for ..."
- Stemming



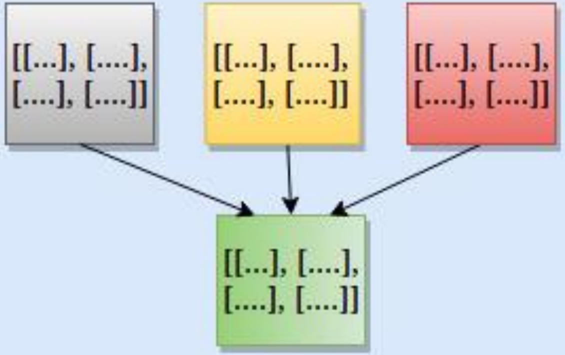
3. Vectors for tools



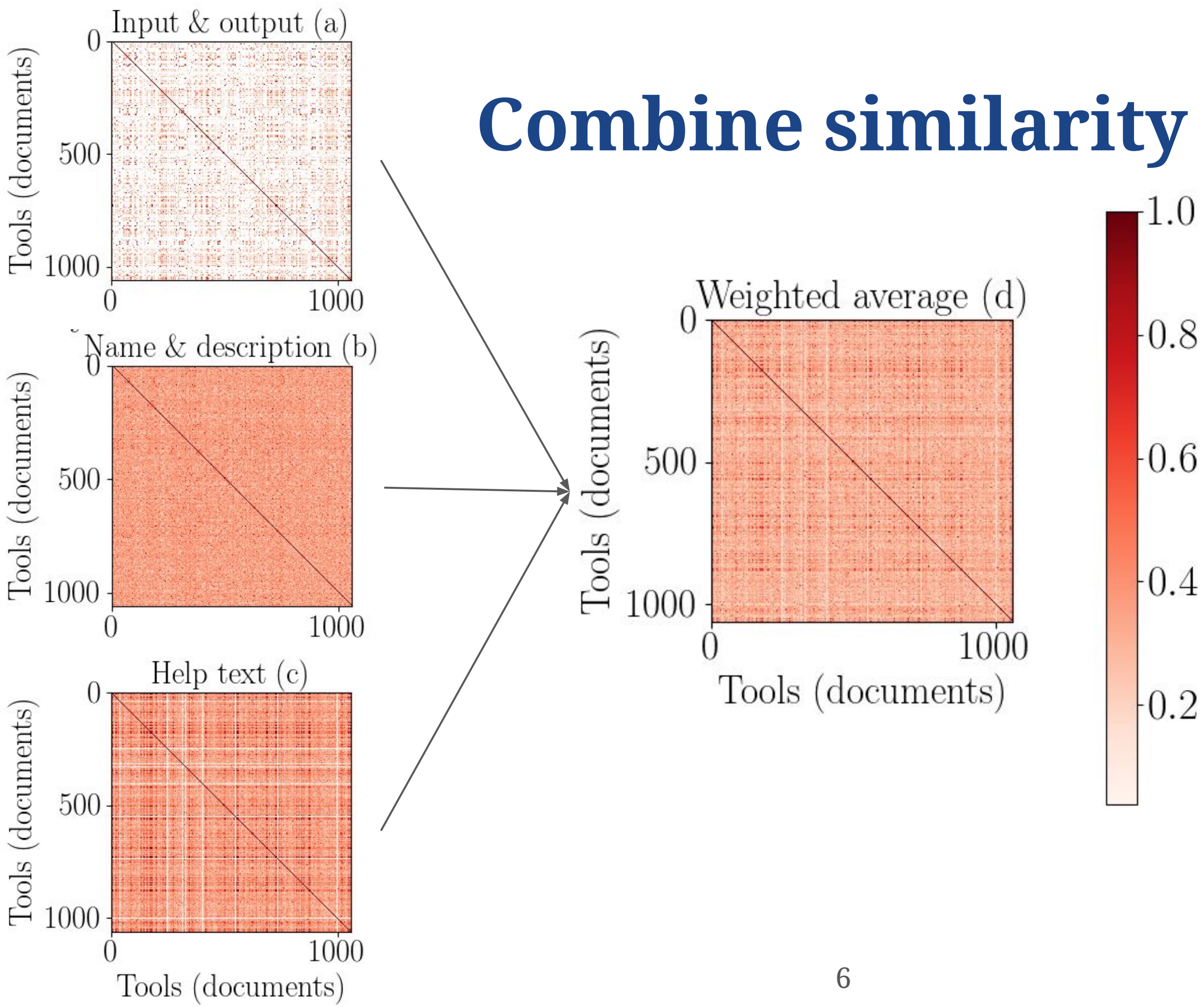
4. Similarity between vectors



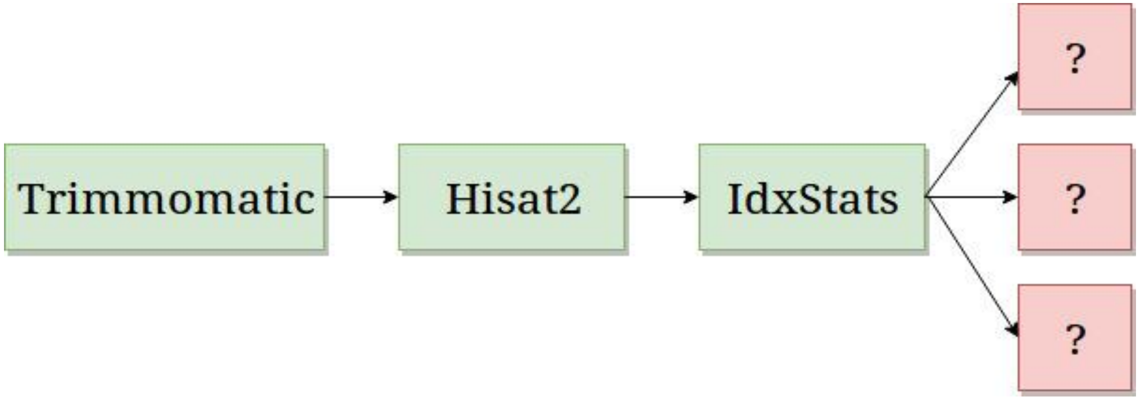
5. Combine similarity matrices



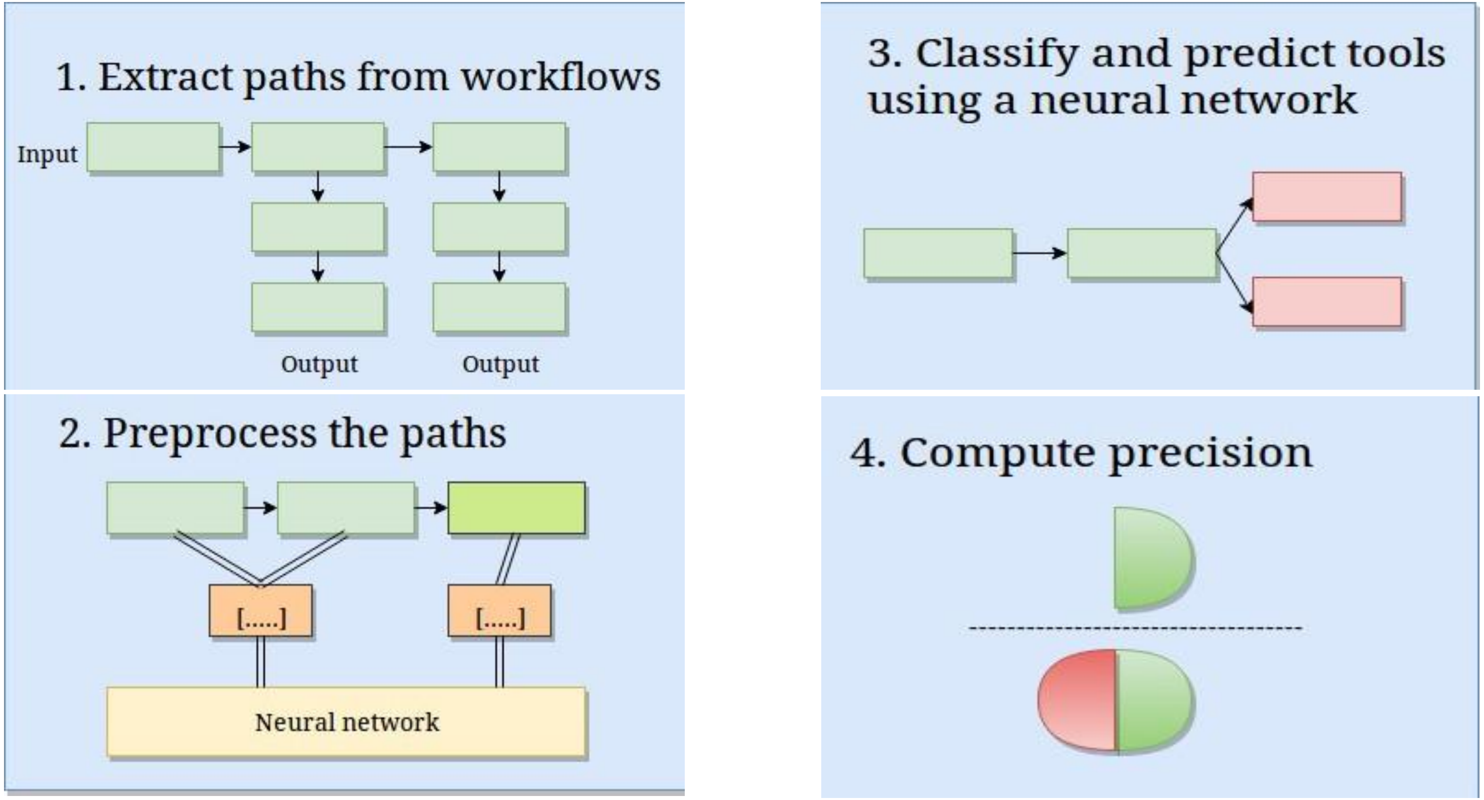
Combine similarity matrices



2. Predict tools in workflows

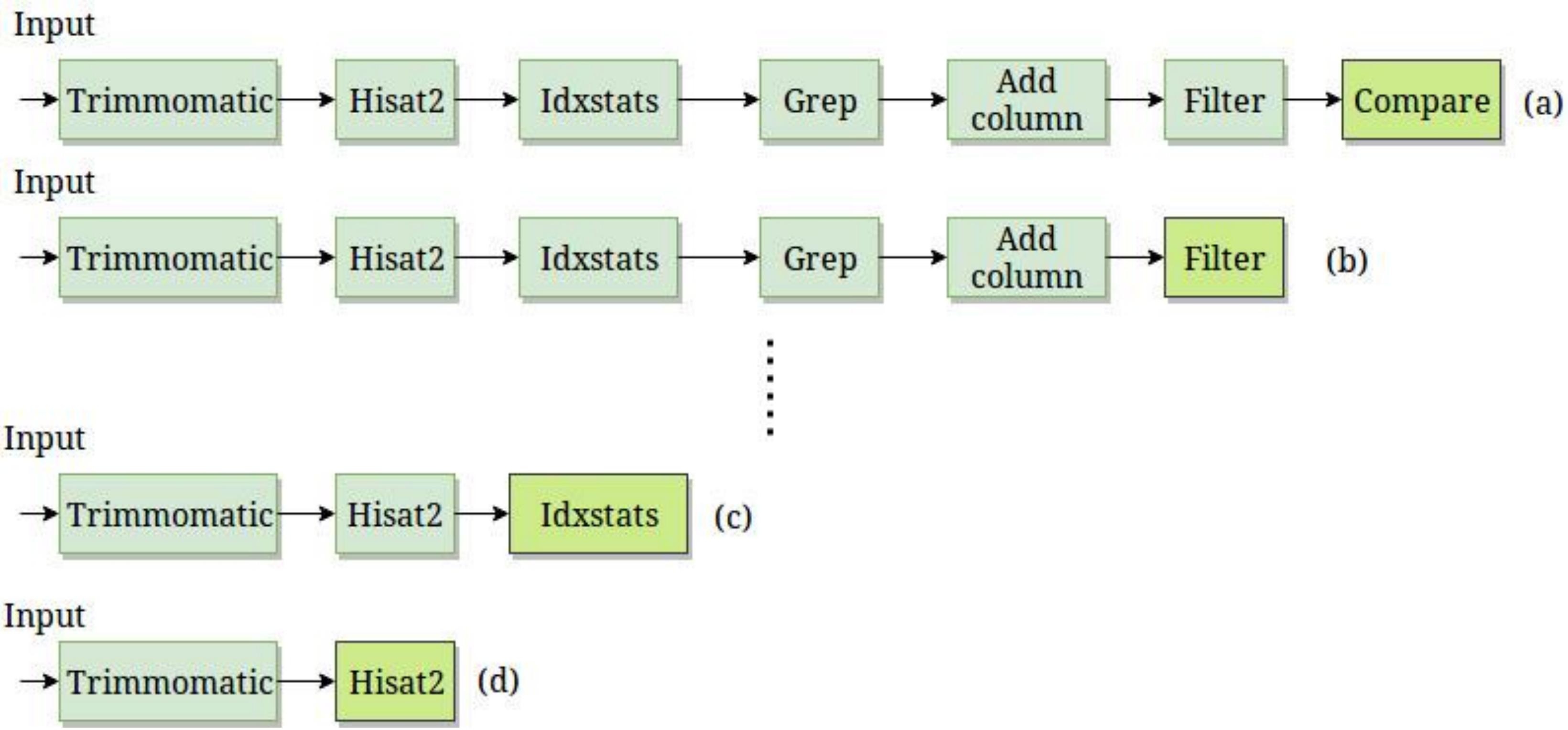


Approach



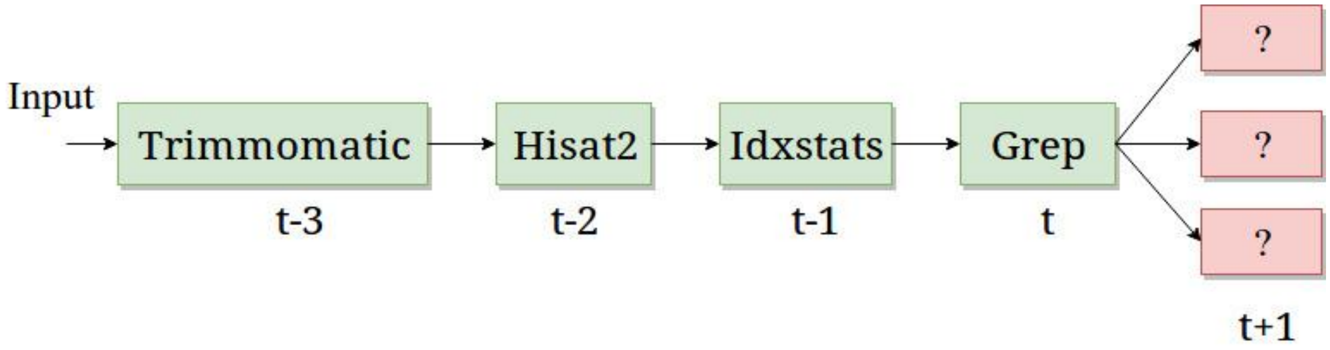
1. Extract paths from workflows
Input
Output
Output
3. Classify and predict tools using a neural network
2. Preprocess the paths
[.....]
[.....]
Neural network
4. Compute precision

Path preprocessing



Recurrent neural network

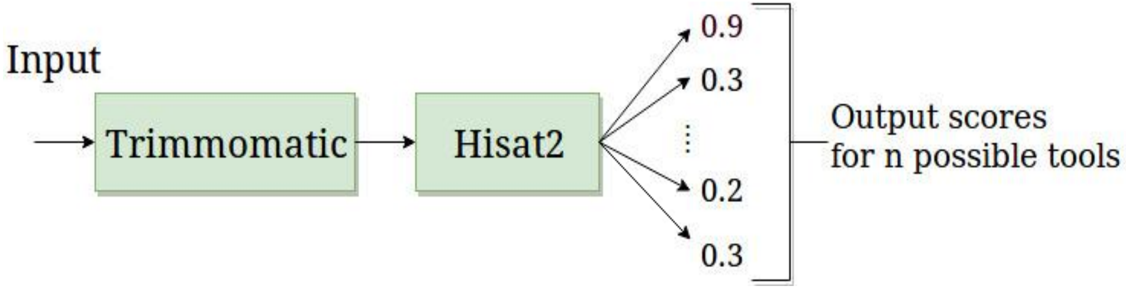
- Classifier to learn on sequential data
- Recurrent neural network - Gated recurrent units



Recurrent neural network: https://arxiv.org/pdf/1412.3555.pdf
Gated recurrent units: https://arxiv.org/pdf/1412.3555v1.pdf

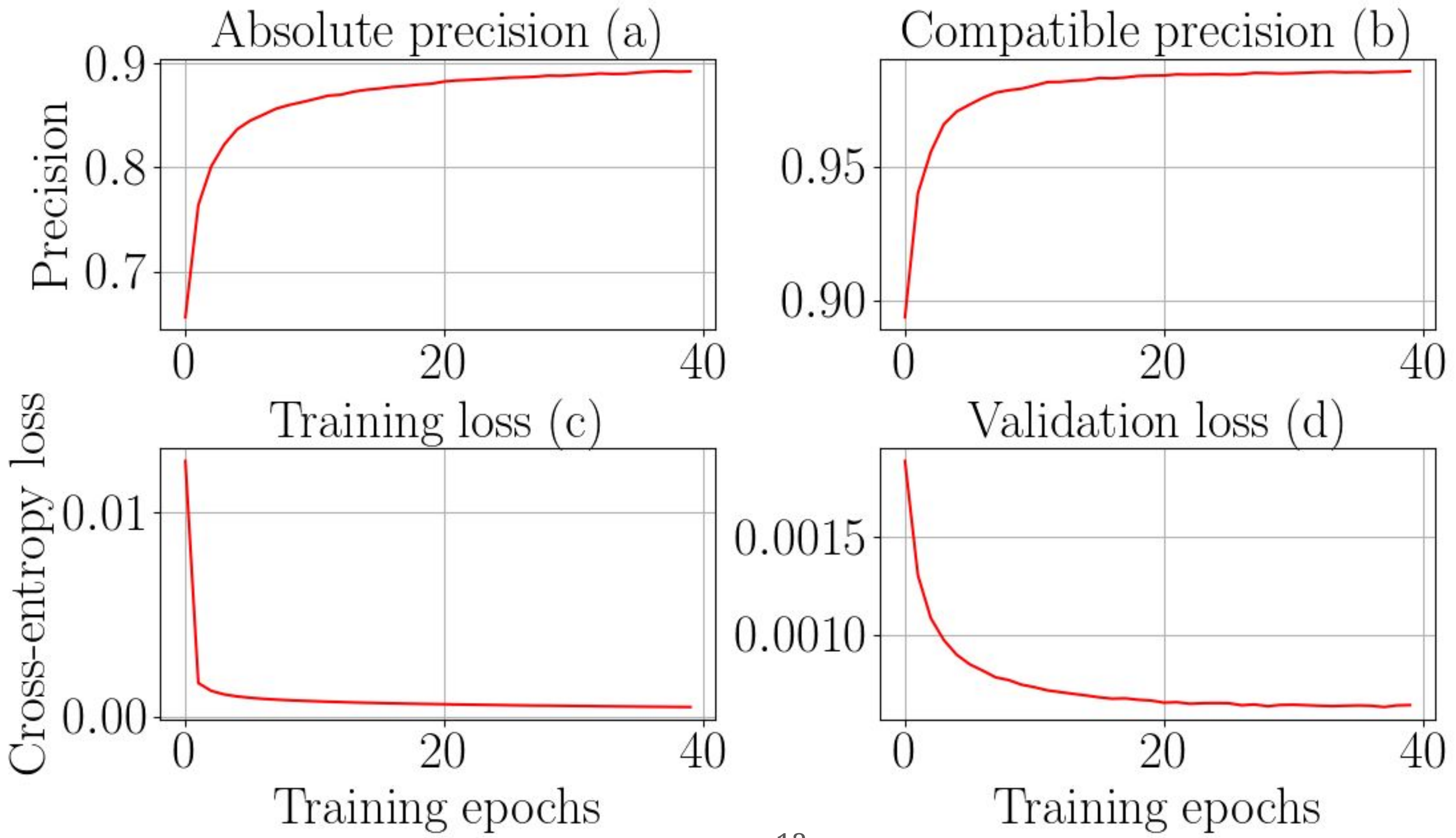
Prediction and precision

- Score for each predicted tool



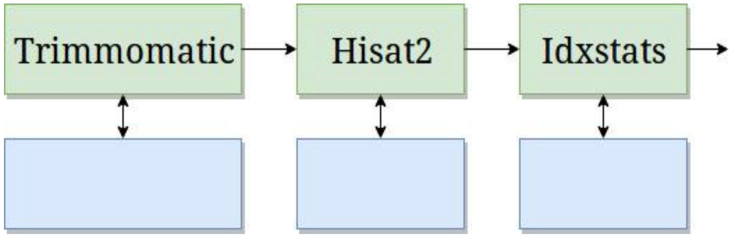
- Absolute precision
- Compatible precision

Precision and loss for decomposition of train and test paths



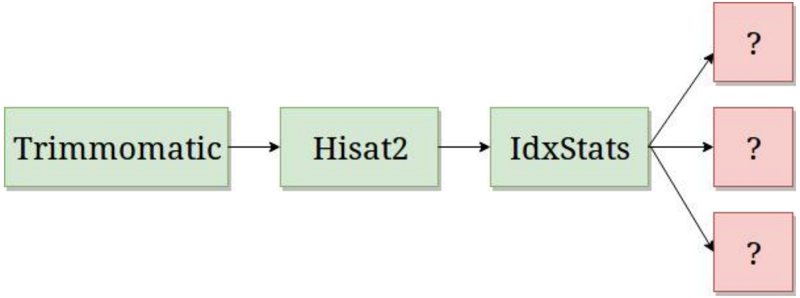
Conclusion and future work (part 1)

- Collect and clean tools metadata (~ 1,050 tools)
- Learn vectors for each tool
- Compute and combine similarity matrices
- Run analysis on larger set of tools
- Compute similar tools using workflows



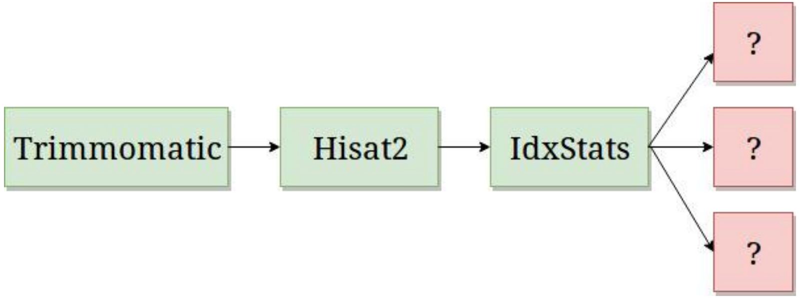
Summary and conclusion (part 2)

- Workflows - directed acyclic graphs (193,000)
- Paths (167,000 unique)
- Recurrent neural network (gated recurrent units)
- Absolute precision ~ 89%, compatible precision ~ 99% (~ 48 hrs)
- More workflows, better precision
- Recommendation system using similar and predicted tools



Future work (part 2)

- Restore original distribution
- Decay prediction over time
- Integrate into Galaxy



Thank you all!

- Prof. Dr. Rolf Backofen
- Prof. Dr. Wolfgang Hess
- Dr. Björn Grüning
- Dr. Anika Erxleben
- Helena Rasche
- Nate Coraor (Galaxy team, Penn State University)
- Freiburg Galaxy team and Bioinformatics Group Freiburg

Thank you for your attention

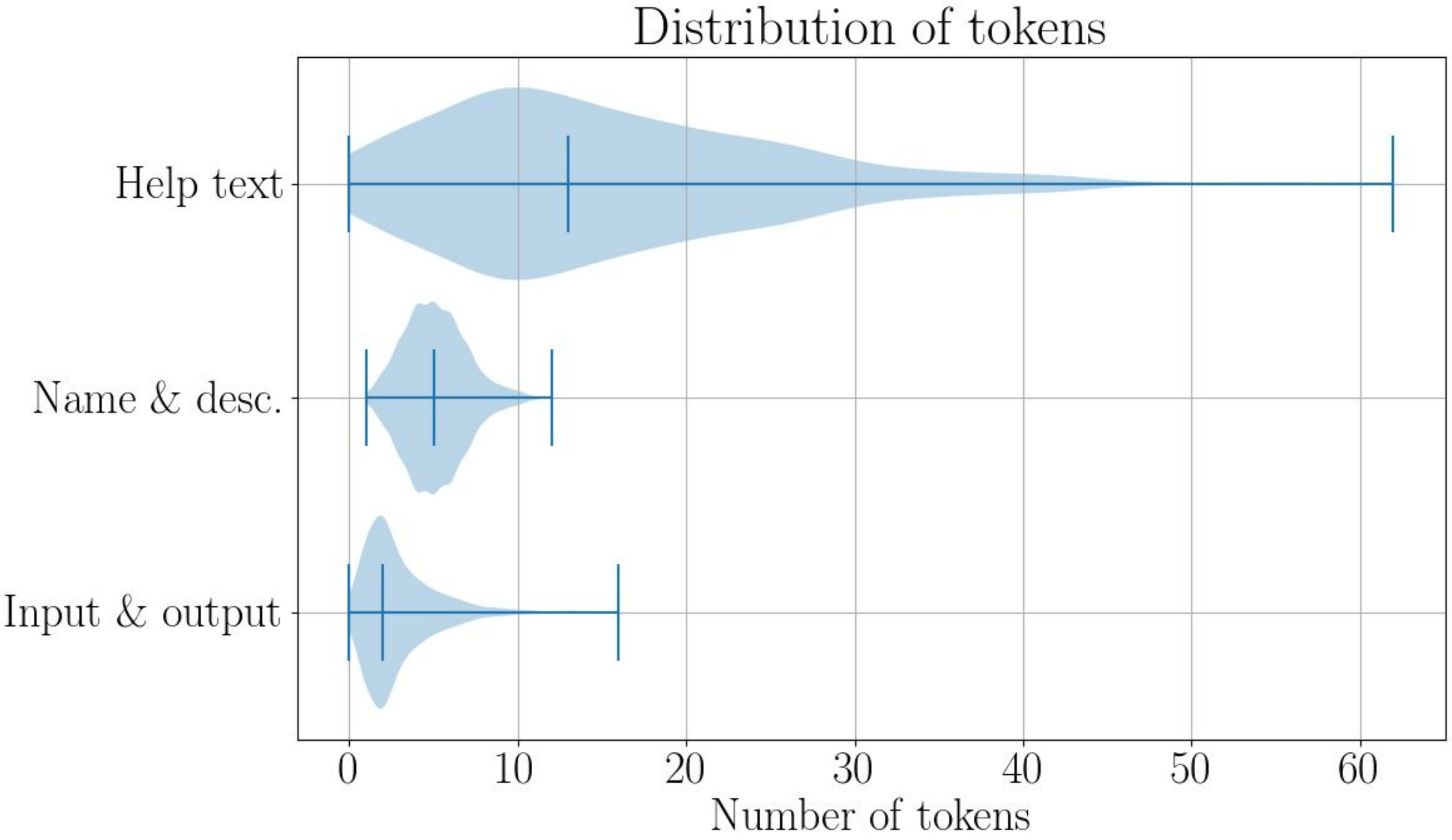
Questions?

Supplementary material

Stemming and stopwords

- Stemming - converge all forms of a word into one basic form
- "*Operate, operating, operates, operation, operative, operatives, operational*" into "*oper*" [1]
- Stopwords - "*a, about, above, would, could …*" [2]

1. https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html
2. https://www.ranks.nl/stopwords


Distribution of tokens
Help text
Name & desc.
Input & output
0
10
20
30
40
50
60
Number of tokens

Bestmatch25 (bm25)

- Token frequency (tf)
- Document and inverted document frequency (idf)

$$idf = \log \frac{N}{df}$$

$$\alpha = (1 - b) + \frac{b \cdot |D|}{|D|_{avg}}$$

$$tf^* = tf \cdot \frac{k+1}{k \cdot \alpha + tf}$$

$$bm25 = tf^* \cdot idf$$

https://dl.acm.org/citation.cfm?id=1704810
http://www.staff.city.ac.uk/~sb317/papers/foundations_bm25_review.pdf

Bestmatch25 (bm25) scores

Tools/Tokens	Regress	Linear	Gap	Mapper	Perform
LinearRegression	5.22	4.1	0.0	0.0	3.84
LogisticRegression	3.54	0.0	0.0	0.0	2.61
Tophat2	0.0	0.0	1.47	1.47	0.0
Hisat	0.0	0.0	0.0	0.0	0.0

Latent Semantic Analysis

- Document-token matrix (*X*)
- Singular value decomposition

$$X_{n\times m} = U_{n\times n} \cdot S_{n\times m} \cdot V_{m\times m}^T$$

$$U^T \cdot U = I_{n\times n}$$

$$V^T \cdot V = I_{m\times m}$$

$$X_{n\times m} = U_k \cdot S_k \cdot V_k^T$$

http://theory.stanford.edu/~tim/s15/l/l9.pdf

Paragraph (document) vector

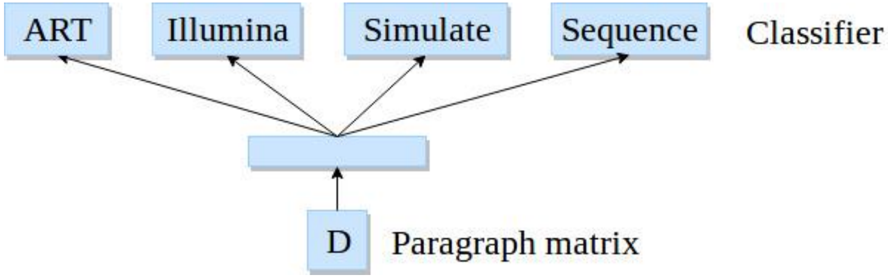
- A dense vector for each paragraph
- Paragraph - a collection of tokens
- Similar paragraphs, similar vectors
- Encode variable length paragraphs
- Each tool has three paragraphs (documents)

Paragraph vector: https://cs.stanford.edu/~quocle/paragraph_vector.pdf
Input/output file types - https://dl.acm.org/citation.cfm?id=1704810 (bestmatch25)

Paragraph vectors

- Softmax classifier
- Backpropagation
- Stochastic gradient descent
- Gensim*
- 800 iterations, 10 epochs

$$\frac{1}{T} \cdot \sum_{t=k}^{T-k} \log p(w_t | w_{t-k}, \ldots, w_{t+k})$$



*https://radimrehurek.com/gensim/models/doc2vec.html
Image adapted from: https://cs.stanford.edu/~quocle/paragraph_vector.pdf

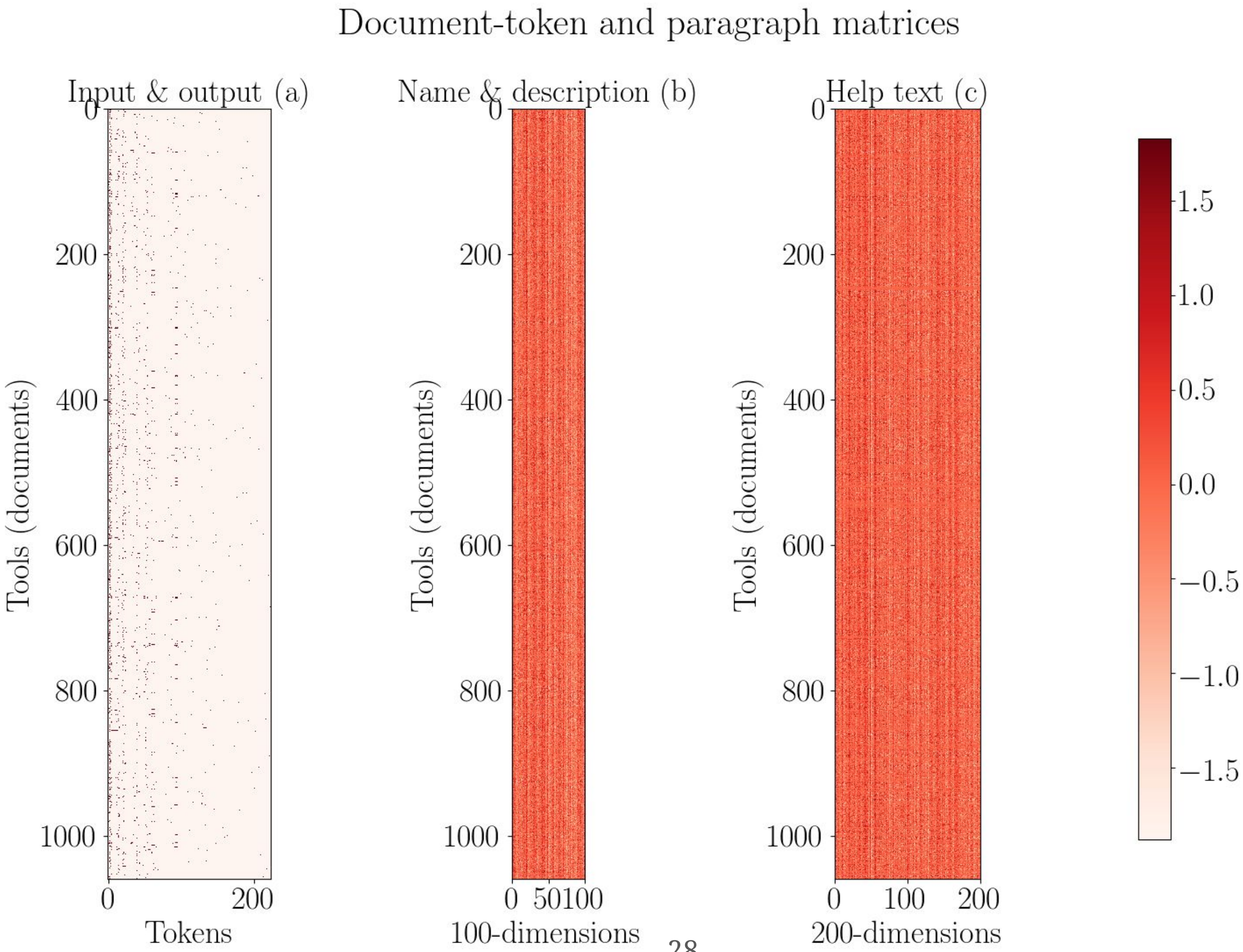
Similarity scores

- Jaccard index (input and output file types)
- Cosine angle (name and description and help text)
- Compute similarity matrix
- Three similarity matrices
- Simple - average the matrices
- Better - learn weights

$$j = \frac{A \cap B}{A \cup B}$$

$$x \cdot y = |x| \times |y| \times \cos\theta$$

Jaccard index: https://link.springer.com/article/10.1007/BF02365362
Cosine angle similarity: https://www.itl.nist.gov/div898/software/dataplot/refman2/auxillar/cosdist.htm


Document-token and paragraph matrices
Input & output (a)
Name & description (b)
Help text (c)
Tools (documents)
Tokens
100-dimensions
200-dimensions
0
200
400
600
800
1000
0 50100
0 100 200
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5

Optimisation

- Gradient descent
- Learn weights on similarity scores
- Mean squared error

$$Error(w^k) = \frac{1}{N} \times \sum_{j=1}^{N} [w^k \times SM^k - SM_{ideal})^2]_j$$

- Each tool, 3 weights
- Obtain a weighted average similarity matrix

True similarity is an array of 1.0
https://developers.google.com/machine-learning/crash-course/reducing-loss/gradient-descent

Optimisation

- Gradient descent
- Learn weights on similarity scores

$$Error(w^k) = \frac{1}{N} \times \sum_{j=1}^{N} [w^k \times SM^k - SM_{ideal})^2]_j \quad \ldots. (1)$$

$$Gradient(w^k) = \frac{\partial Error}{\partial w^k} = \frac{2}{N} \times ((w^k \times SM^k - SM_{ideal}) \cdot SM^k) \quad \ldots. (2)$$

$$w^k = w^k - \eta \times Gradient(w^k) \quad \ldots. (3)$$

- Obtain a weighted average similarity matrix

Optimisation

$$Error_{io}(w_{io}^k) = \frac{1}{N} \times \sum_{j=1}^{N} [(w_{io}^k \times SM_{io}^k - SM_{ideal})^2]_j$$

$$Error_{nd}(w_{nd}^k) = \frac{1}{N} \times \sum_{j=1}^{N} [(w_{nd}^k \times SM_{nd}^k - SM_{ideal})^2]_j$$

$$Error_{ht}(w_{ht}^k) = \frac{1}{N} \times \sum_{j=1}^{N} [(w_{ht}^k \times SM_{ht}^k - SM_{ideal})^2]_j$$

$$Error(w^k) = Error_{io}(w_{io}^k) + Error_{nd}(w_{nd}^k) + Error_{ht}(w_{ht}^k)$$

$$argmin_{w^k} Error(w^k)$$

$$w_{io}^k + w_{nd}^k + w_{ht}^k = 1$$

$$SM^k = w_{io}^k \cdot SM_{io}^k + w_{nd}^k \cdot SM_{nd}^k + w_{ht}^k \cdot SM_{ht}^k$$

Optimisation

1	0.34	0.65	0.44
0.34	1	...	...
0.65	...	1	...
0.44	...	...	1

1	0.76	0.63	0.85
0.76	1	...	...
0.63	...	1	...
0.85	...	...	1

1	0.06	0.1	0.17
0.06	1	...	...
0.1	...	1	...
0.17	...	...	1

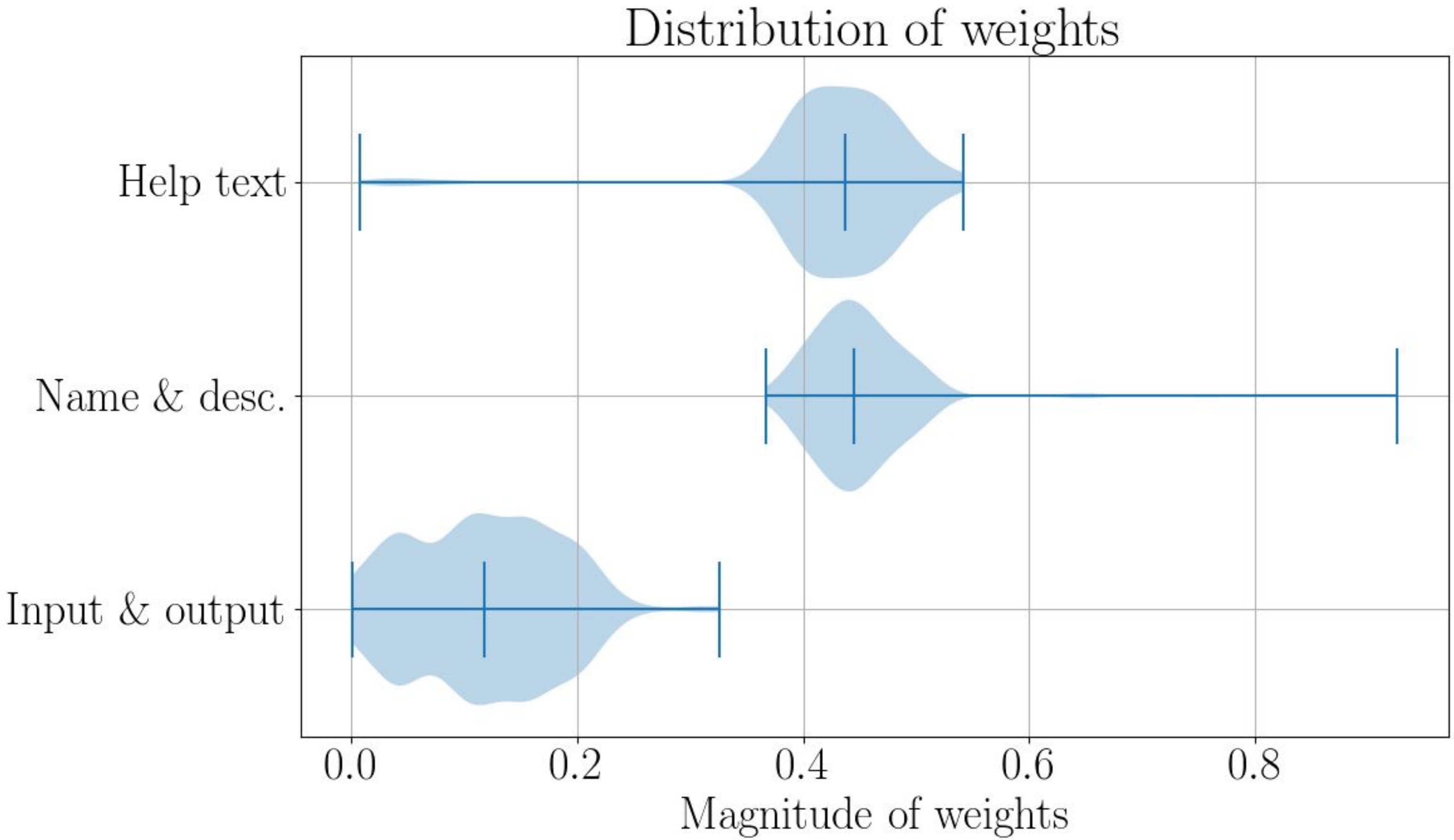
Optimisation

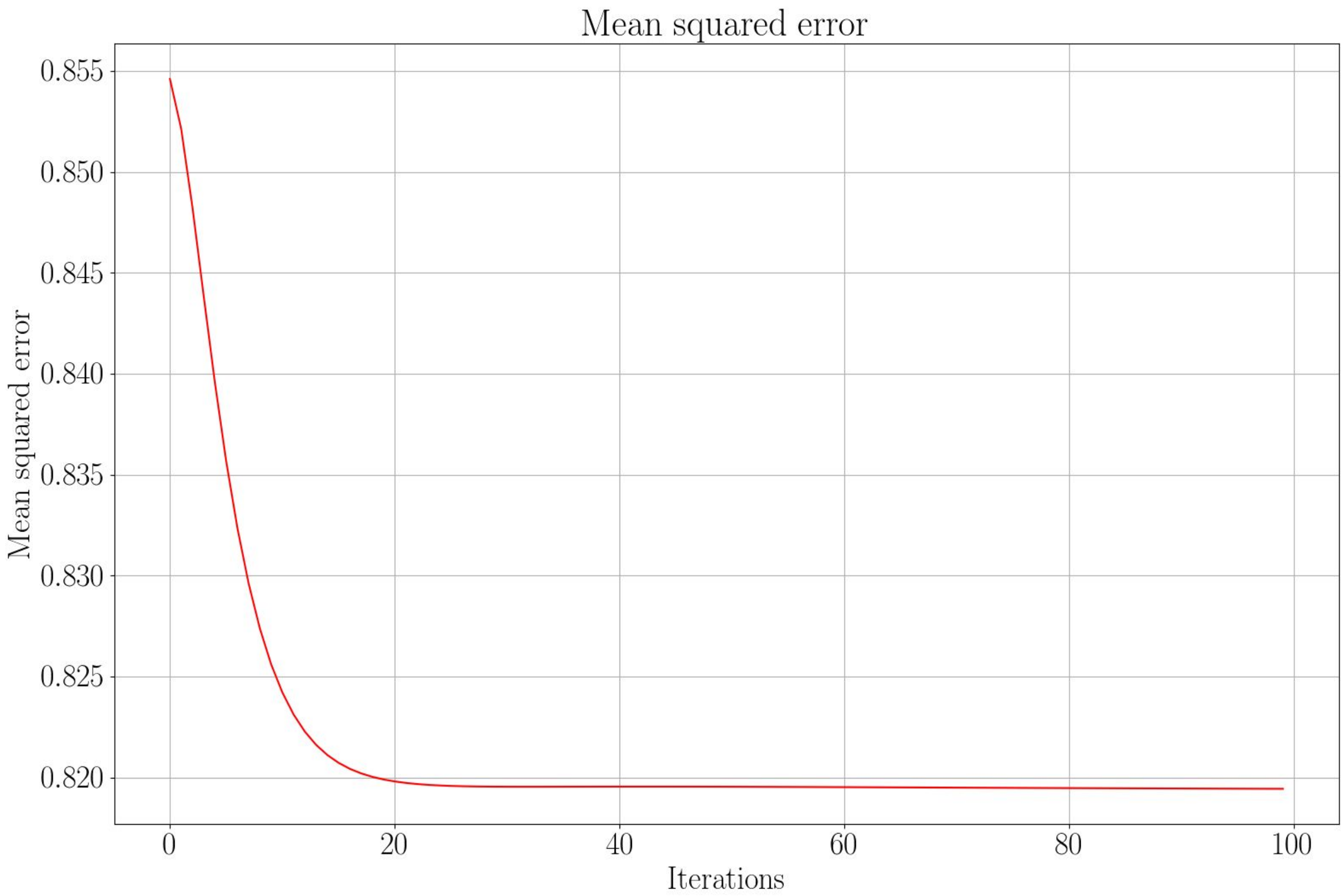
- Initial learning rate - 0.05
- Decay learning rate
- Nesterov's accelerated gradient

$$update_{t+1} = \gamma \cdot update_t - \eta \cdot Gradient(w_t + \gamma \cdot update_t)$$

$$w_{t+1} = w_t + update_{t+1}$$

https://arxiv.org/pdf/1609.04747.pdf


Distribution of weights
Help text
Name & desc.
Input & output
0.0
0.2
0.4
0.6
0.8
Magnitude of weights


Mean squared error
Mean squared error
0.855
0.850
0.845
0.840
0.835
0.830
0.825
0.820
0
20
40
60
80
100
Iterations

Visualisers

- Paragraph vectors
- Latent Semantic Analysis (5% of full-rank)

Workflows

- Workflow - a directed acyclic graph
- ~ 193,000 workflows
- ~ 900,000 paths (~ 167,000 unique)
- Maximum 25 tools in a path

Directed acyclic graph:
https://cran.r-project.org/web/packages/ggdag/vignettes/intro-to-dags.html
https://galaxyproject.org/learn/advanced-workflow/

Paths statistics

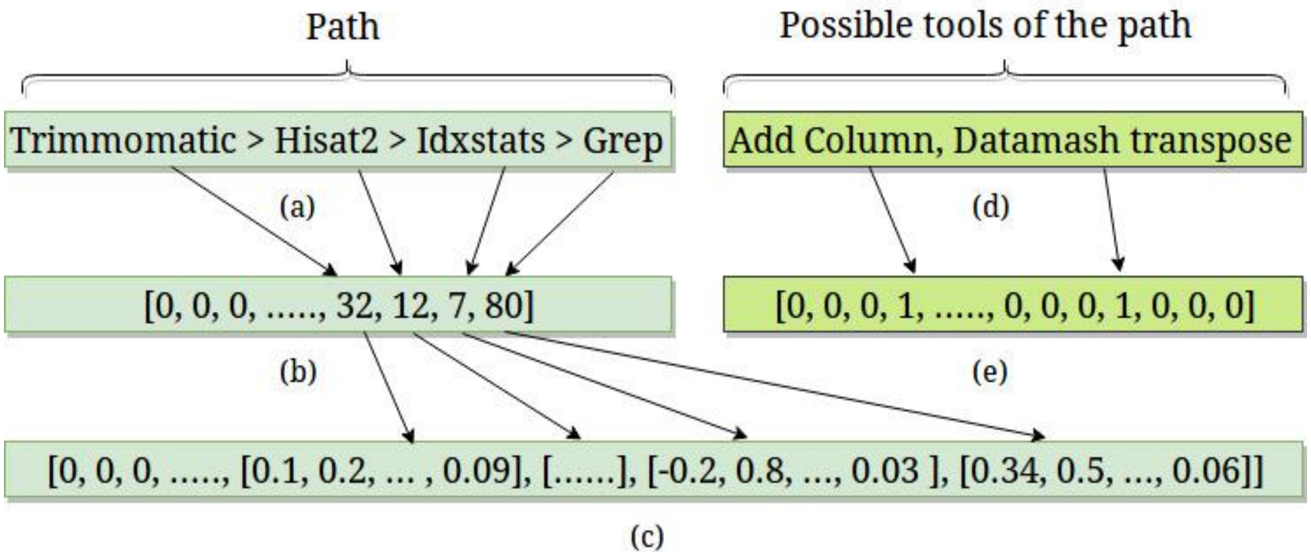
- No decomposition:

  Total paths (111,386), train paths (89,109) and test paths (22,277).

- Decomposition:

  Total paths (210,983), train paths (168,787) and test paths (42,196).

Embedding and label vectors



Dimensions:

- Embedding - 512
- Label vector - 1,800
- Tool sequence - 25

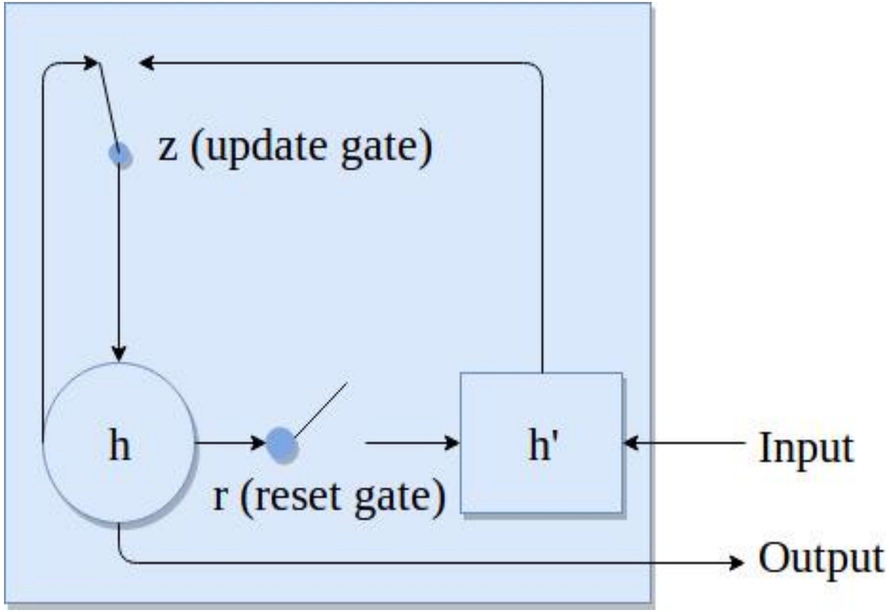
Embedding layer: https://keras.io/layers/embeddings/

Network configuration

- Cross-entropy loss
- Root mean square propagation (rmsprop) optimiser
- 80% training paths, 20% test paths
- 20% of training paths as validation paths
- 1 Embedding layer, 2 hidden layers and 1 output layer

Cross-entropy: http://www.cse.unsw.edu.au/~billw/cs9444/crossentropy.html
RMSProp: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

Recurrent neural network



$$h_t = (1 - z_t) \times h_{t-1} + z_t \times h'_t$$

$$z_t = \sigma(W_z \times x_t + U_z \times h_{t-1})$$

$$r_t = \sigma(W_r \times x_t + U_r \times h_{t-1})$$

$$h'_t = \tanh(W \times x_t + U \times (r_t \odot h_{t-1})$$

$$p(x_T | x_1, x_2, \ldots, x_{T-1})$$

RNN: https://arxiv.org/pdf/1412.3555.pdf

Recurrent neural network

- One embedding layer, two hidden layer and one output layer
- Recurrent layer activation - exponential linear unit

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha \times (e^{x} - 1), & \text{if } x \leq 0, \alpha > 0 \end{cases}$$

- Output activation is sigmoid $f(x) = \frac{1}{1+e^{-x}}$

Exponential linear units: https://arxiv.org/pdf/1511.07289.pdf

Recurrent neural network

- Optimiser - Root mean square propagation (rmsprop)

$$MeanSquare(w,t) = 0.9 \times MeanSquare(w,t-1) + 0.1 \times (\frac{\partial E}{\partial w}(t))^2$$

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{MeanSquare(w,t)+\epsilon}} \times \frac{\partial E}{\partial w}(t)$$

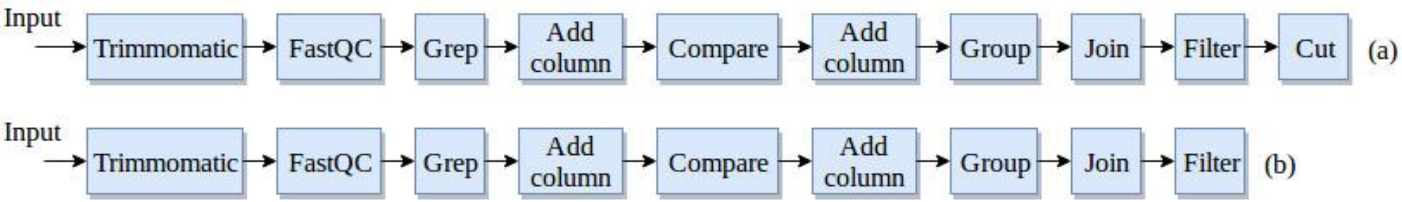
- Binary cross-entropy loss

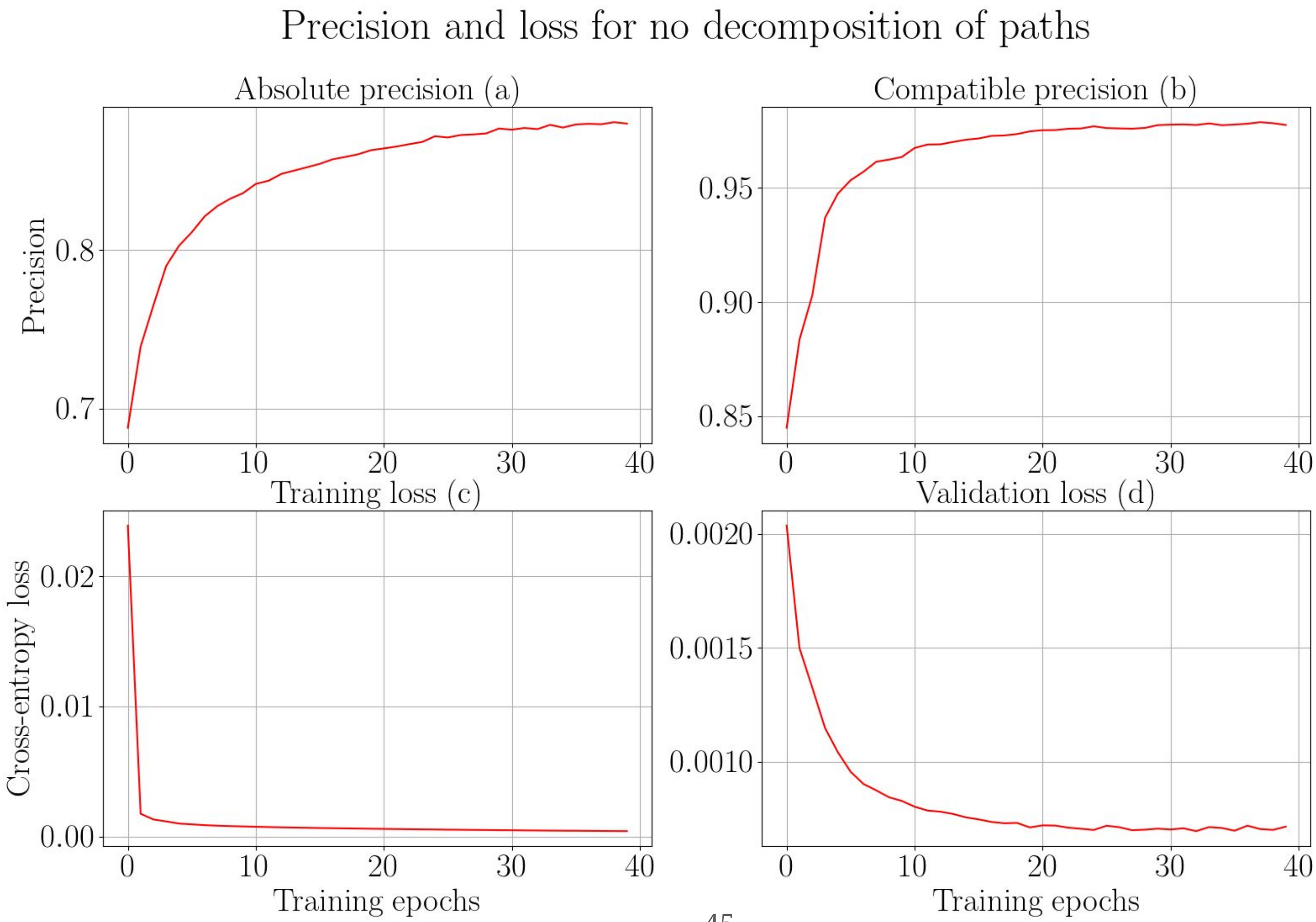
$$loss_{mean} = -\frac{1}{N}(\sum_{i=1}^{N} y_i \times log(p_i) + (1-y_i) \times log(1-p_i))$$

RMSProp: https://arxiv.org/pdf/1609.04747.pdf
Binary cross-entropy: https://www.tensorflow.org/api_docs/python/tf/keras/losses/binary_crossentropy

Workflow preprocessing

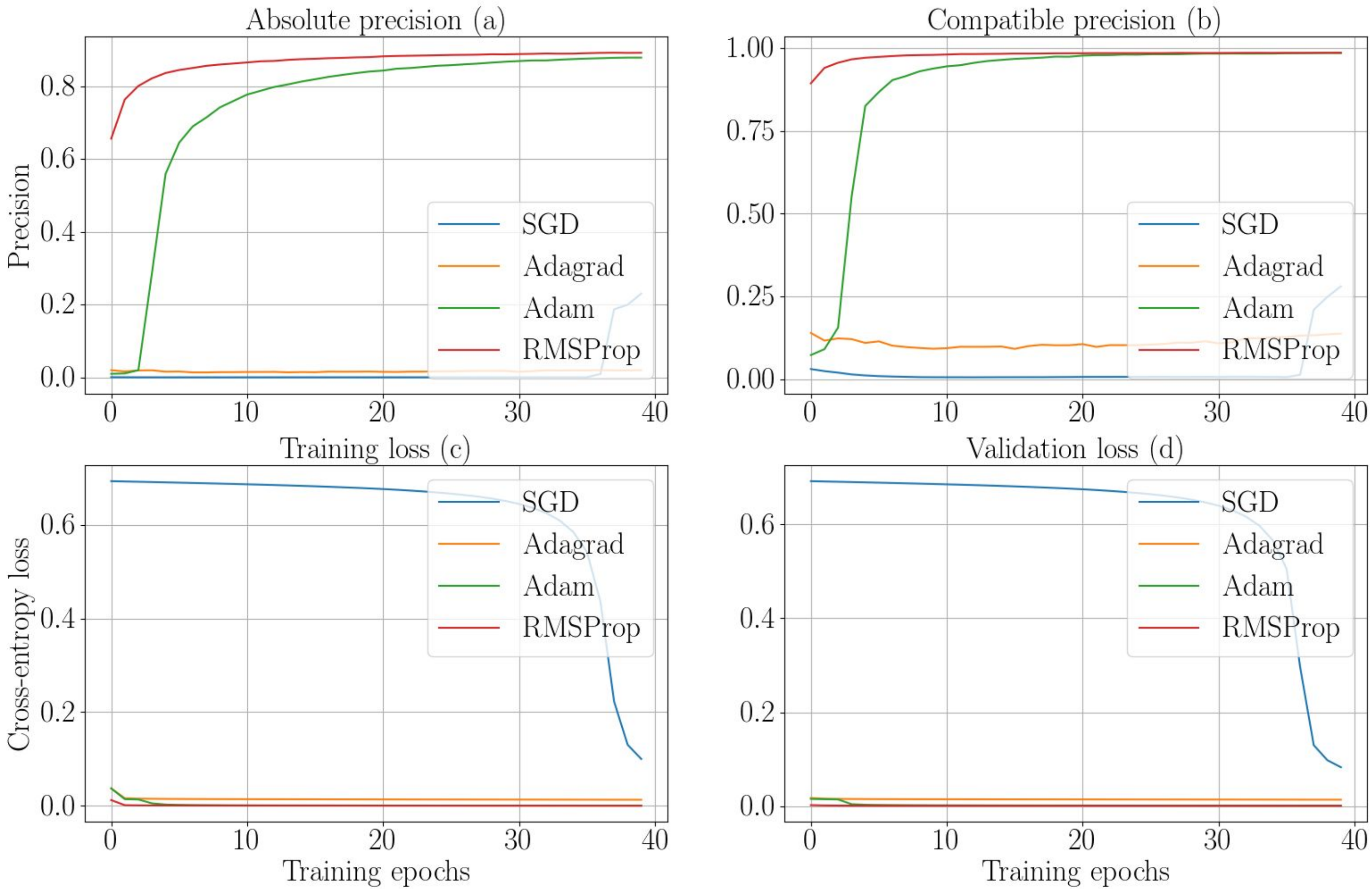
- No decomposition
- Last tool is a label



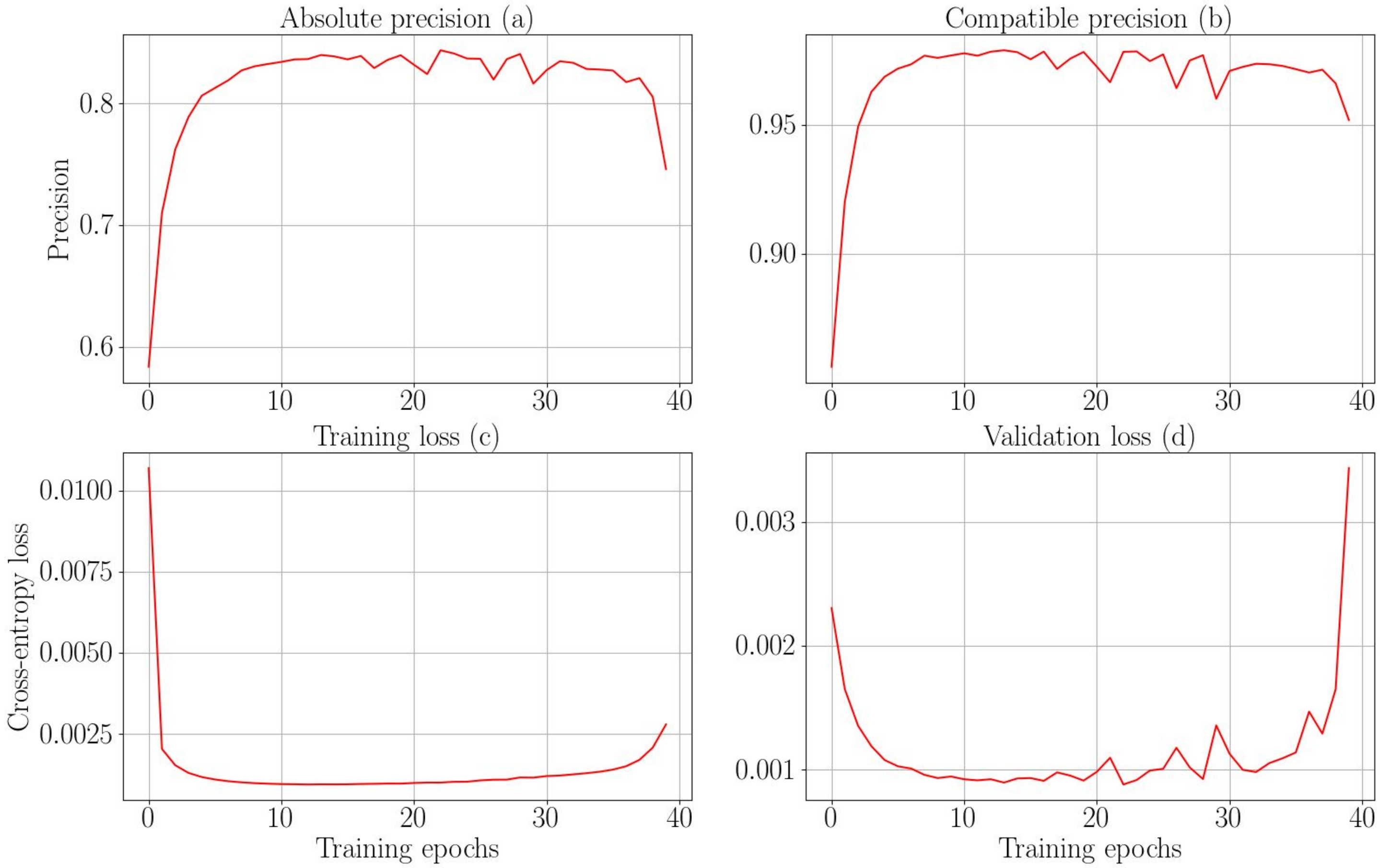


Precision and loss for no decomposition of paths
Absolute precision (a)
Compatible precision (b)
Training loss (c)
Validation loss (d)
Precision
Cross-entropy loss
Training epochs
Training epochs
0.7
0.8
0.85
0.90
0.95
0.00
0.01
0.02
0.0010
0.0015
0.0020
0
10
20
30
40

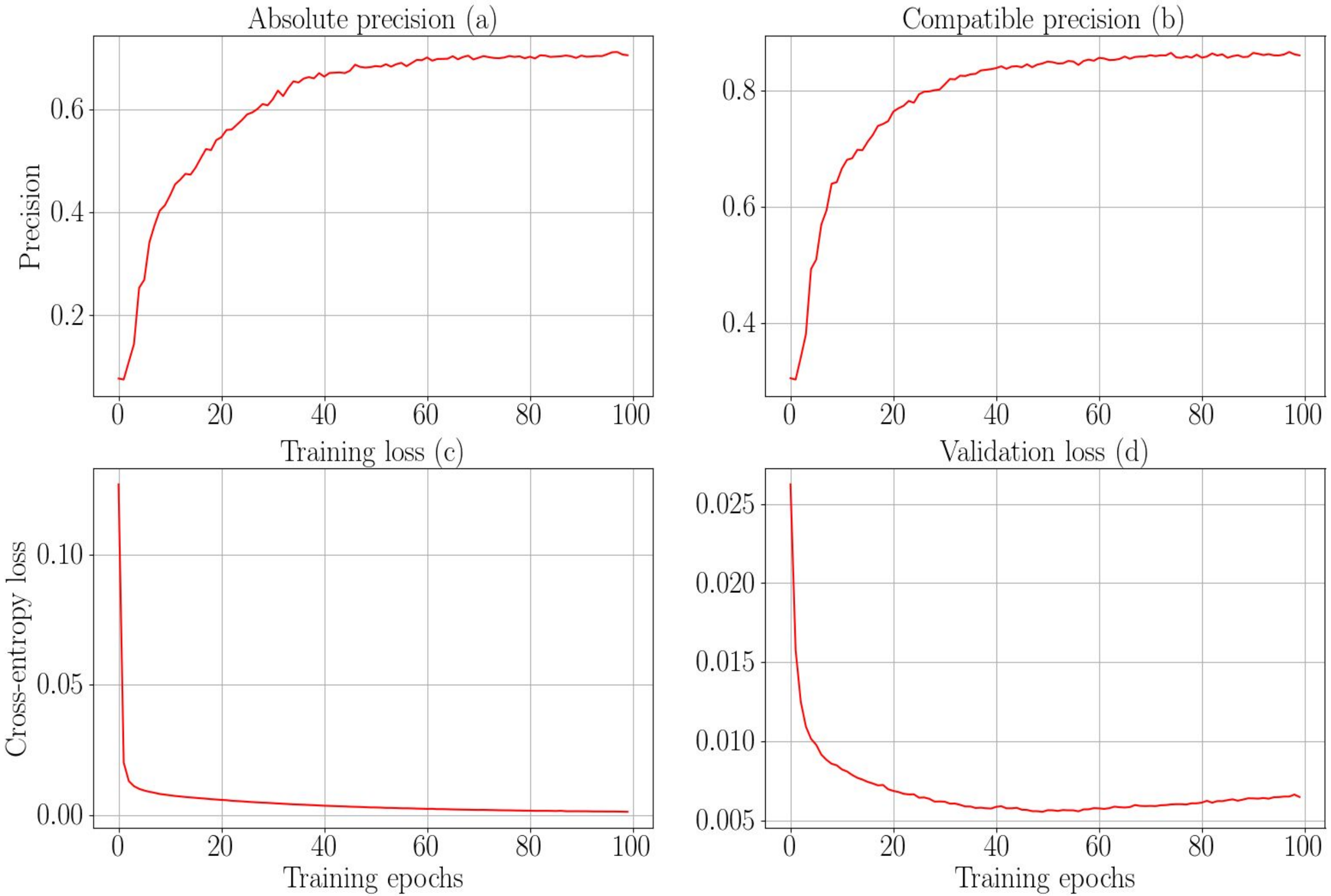
Precision and loss for various optimisers

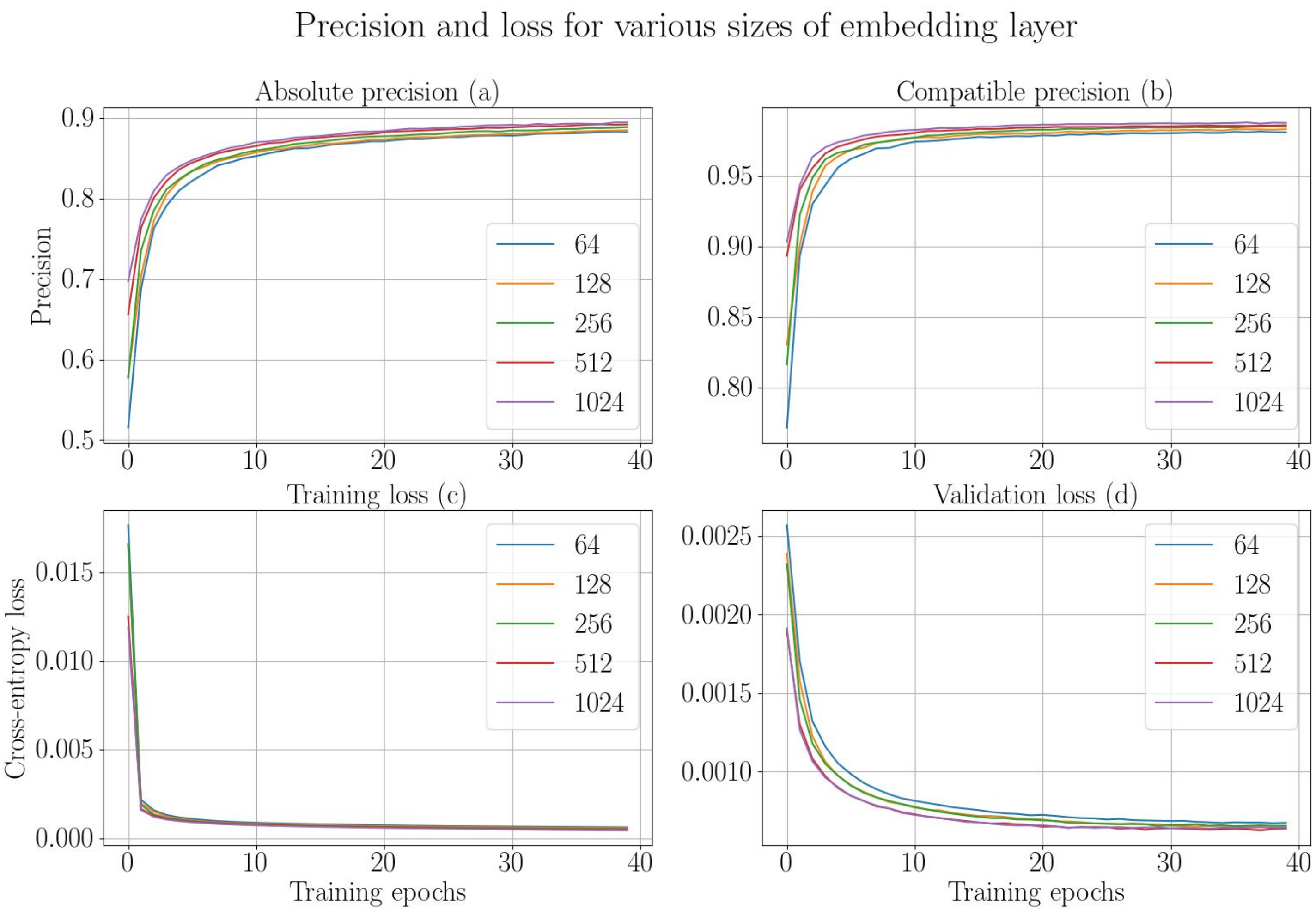


Precision and loss using neural network with dense layers

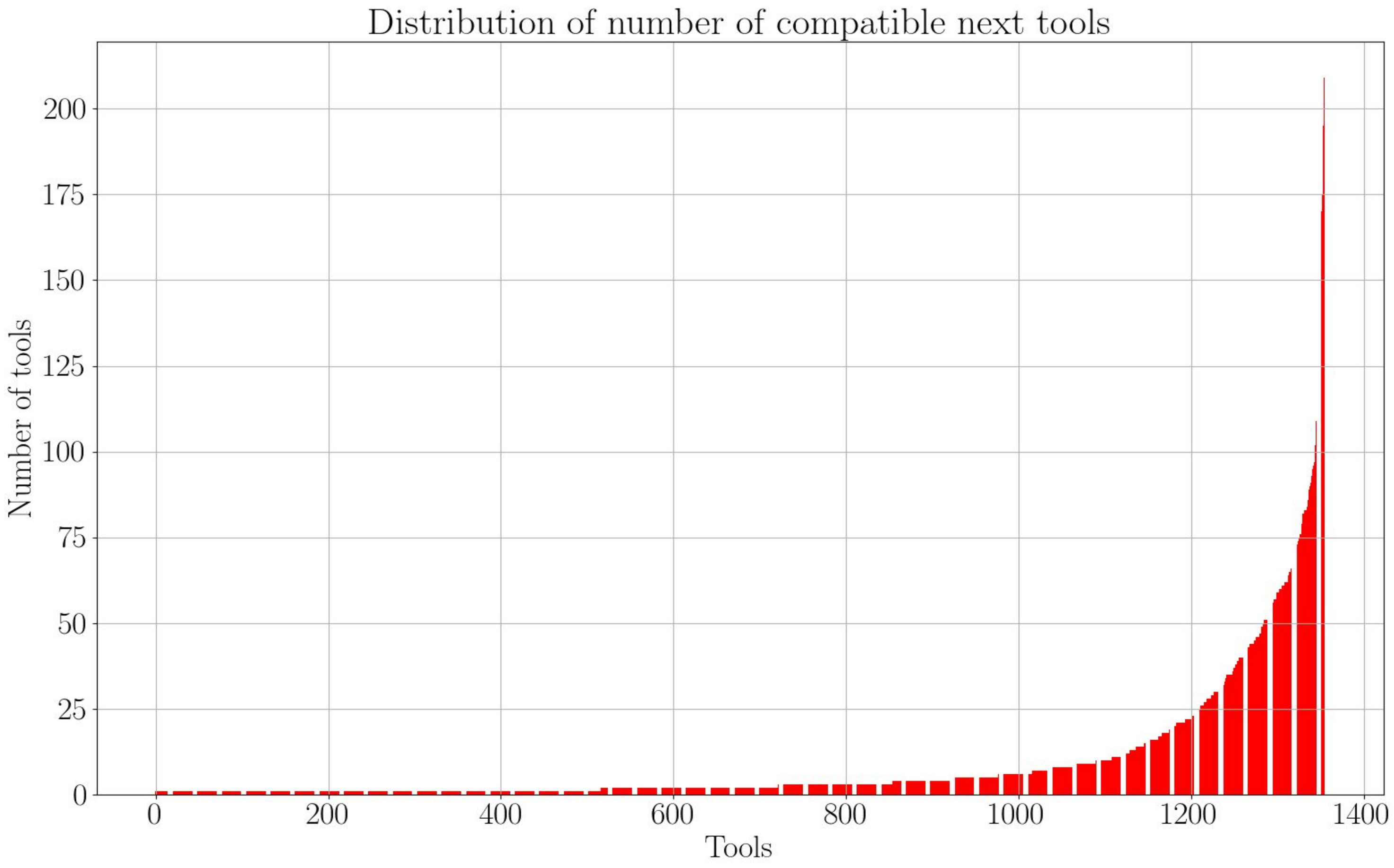


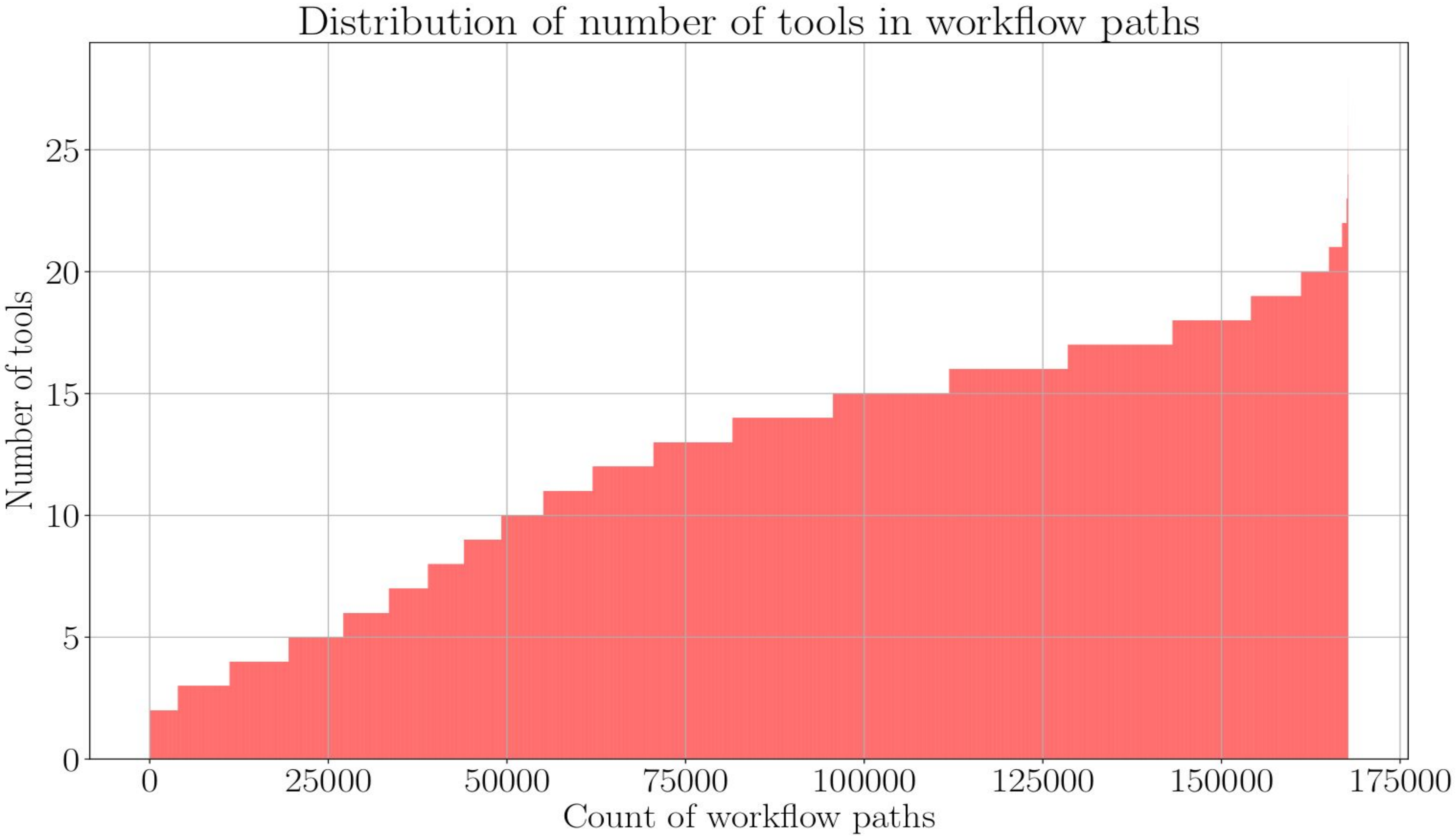
Precision and loss using less data

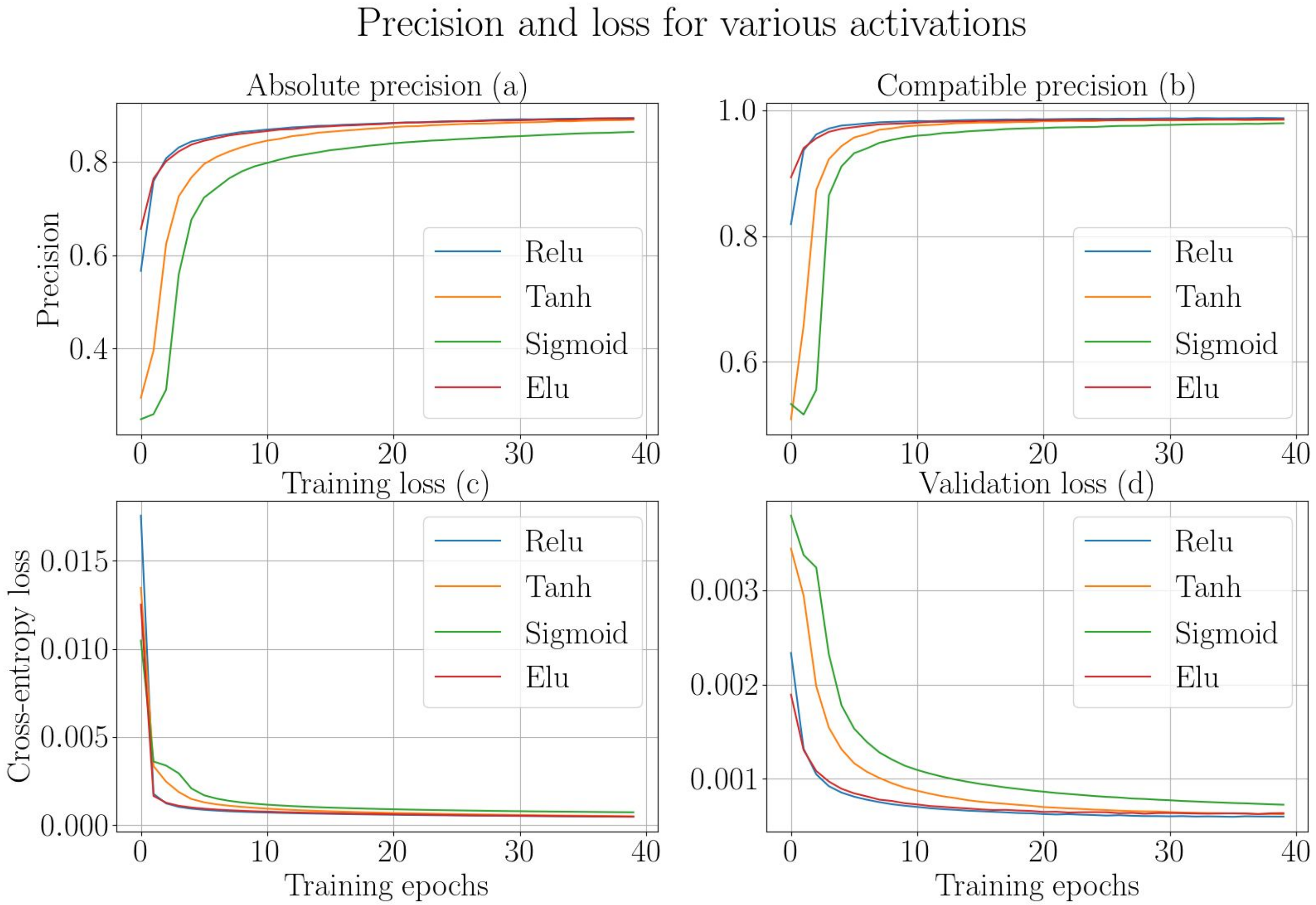


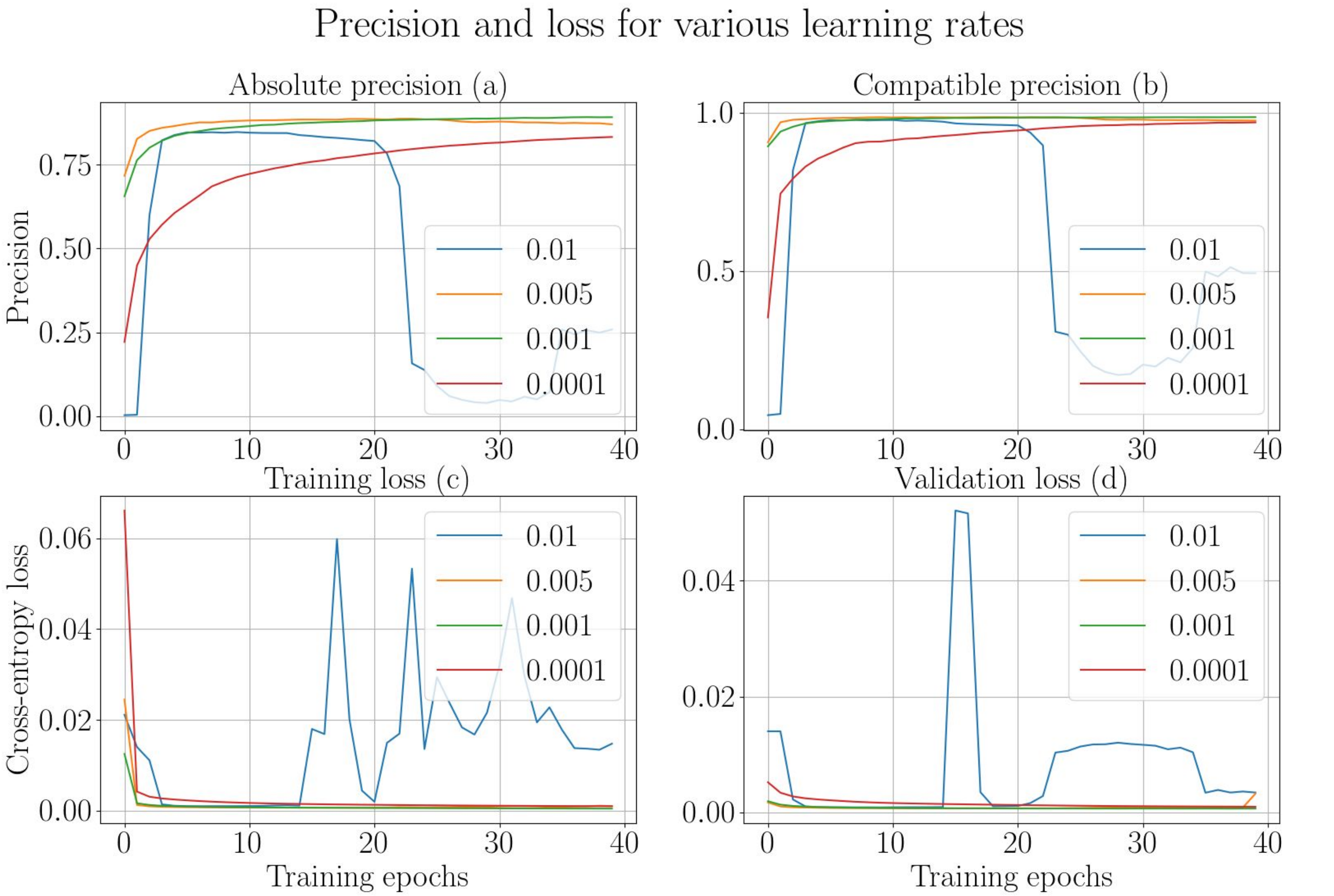


Precision and loss for various sizes of embedding layer
Absolute precision (a)
Compatible precision (b)
Training loss (c)
Validation loss (d)
Precision
Cross-entropy loss
Training epochs
64
128
256
512
1024

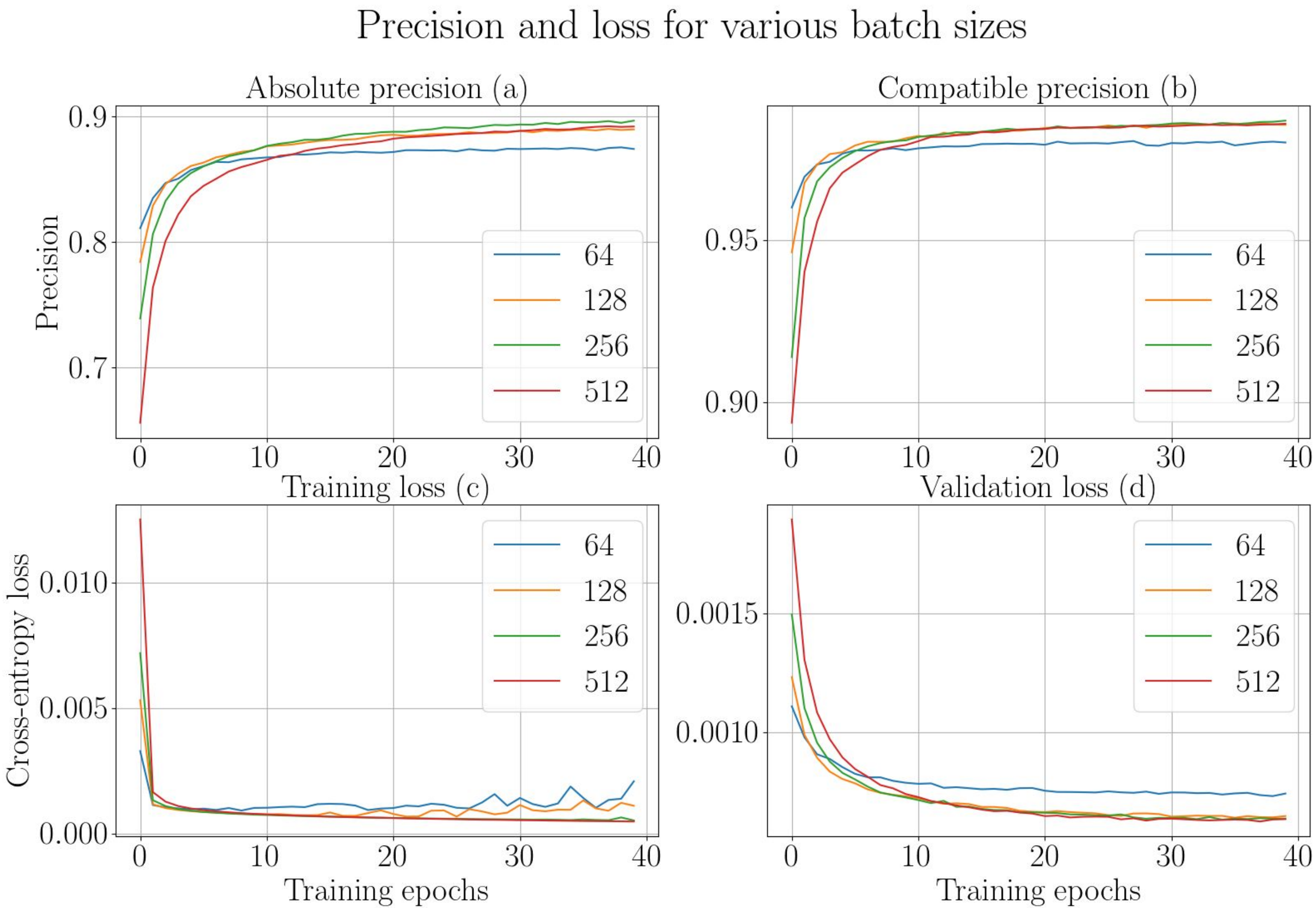

Distribution of number of compatible next tools
Number of tools
0
25
50
75
100
125
150
175
200
0
200
400
600
800
1000
1200
1400
Tools

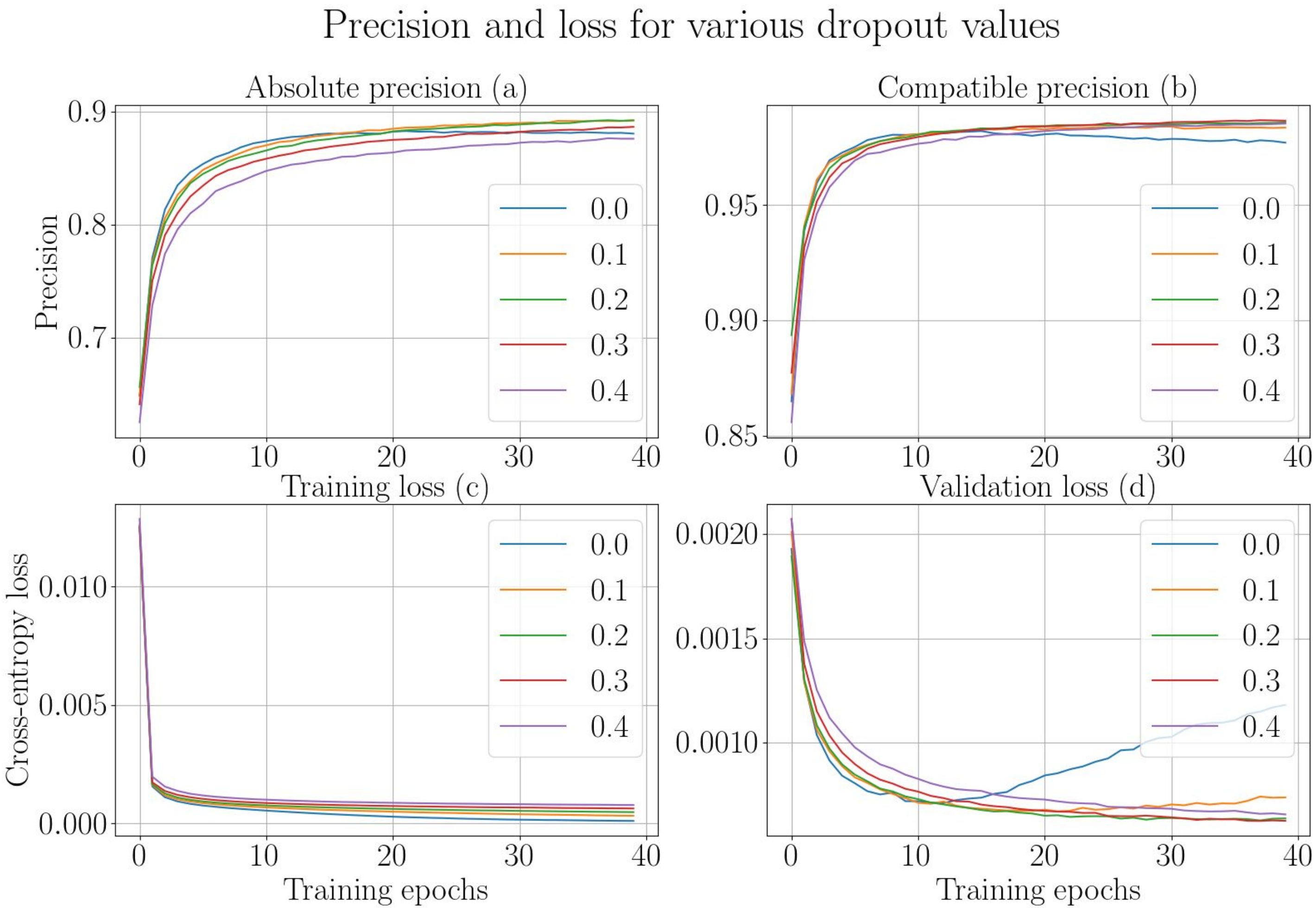

Distribution of number of tools in workflow paths
Number of tools
Count of workflow paths
0
5
10
15
20
25
0
25000
50000
75000
100000
125000
150000
175000


Precision and loss for various activations
Absolute precision (a)
Precision
0.4
0.6
0.8
0
10
20
30
40
Relu
Tanh
Sigmoid
Elu
Compatible precision (b)
0.6
0.8
1.0
0
10
20
30
40
Relu
Tanh
Sigmoid
Elu
Training loss (c)
Cross-entropy loss
0.000
0.005
0.010
0.015
0
10
20
30
40
Training epochs
Relu
Tanh
Sigmoid
Elu
Validation loss (d)
0.001
0.002
0.003
0
10
20
30
40
Training epochs
Relu
Tanh
Sigmoid
Elu

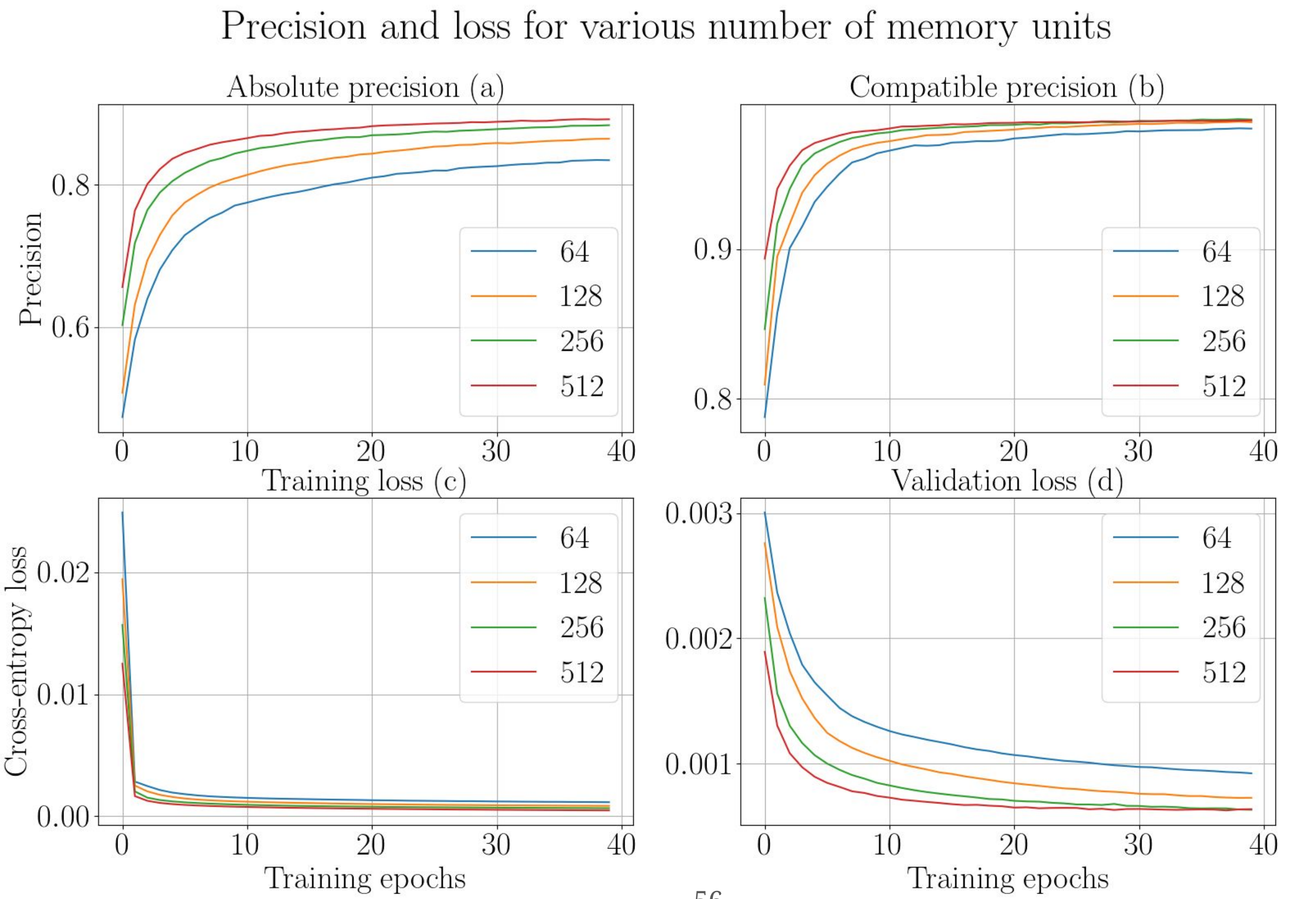


Precision and loss for various learning rates
Absolute precision (a)
Compatible precision (b)
Training loss (c)
Validation loss (d)
Precision
Cross-entropy loss
Training epochs
0.01
0.005
0.001
0.0001

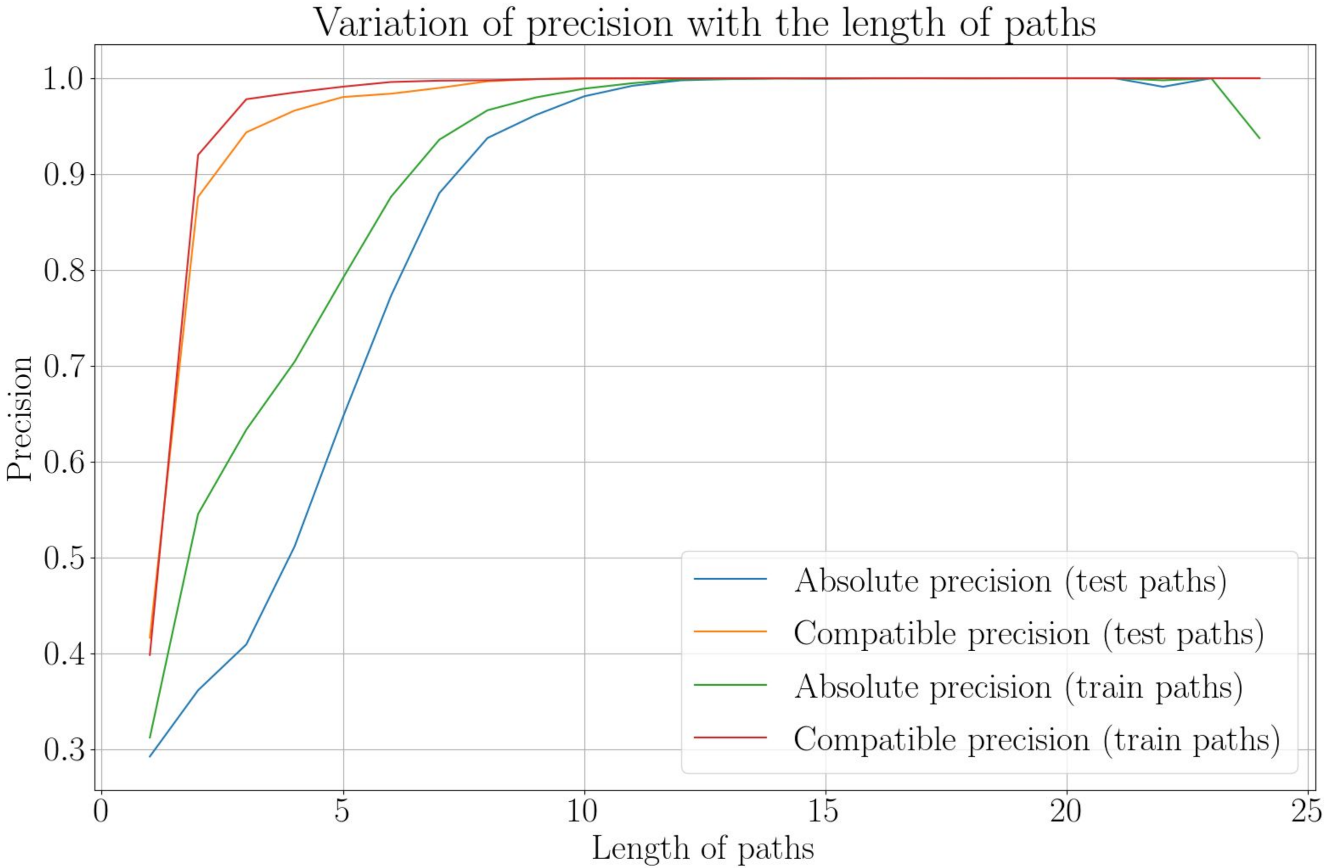

Precision and loss for various batch sizes
Absolute precision (a)
Compatible precision (b)
Training loss (c)
Validation loss (d)
Precision
Cross-entropy loss
Training epochs
Training epochs
64
128
256
512
0.9
0.8
0.7
0.95
0.90
0.010
0.005
0.000
0.0015
0.0010
0
10
20
30
40



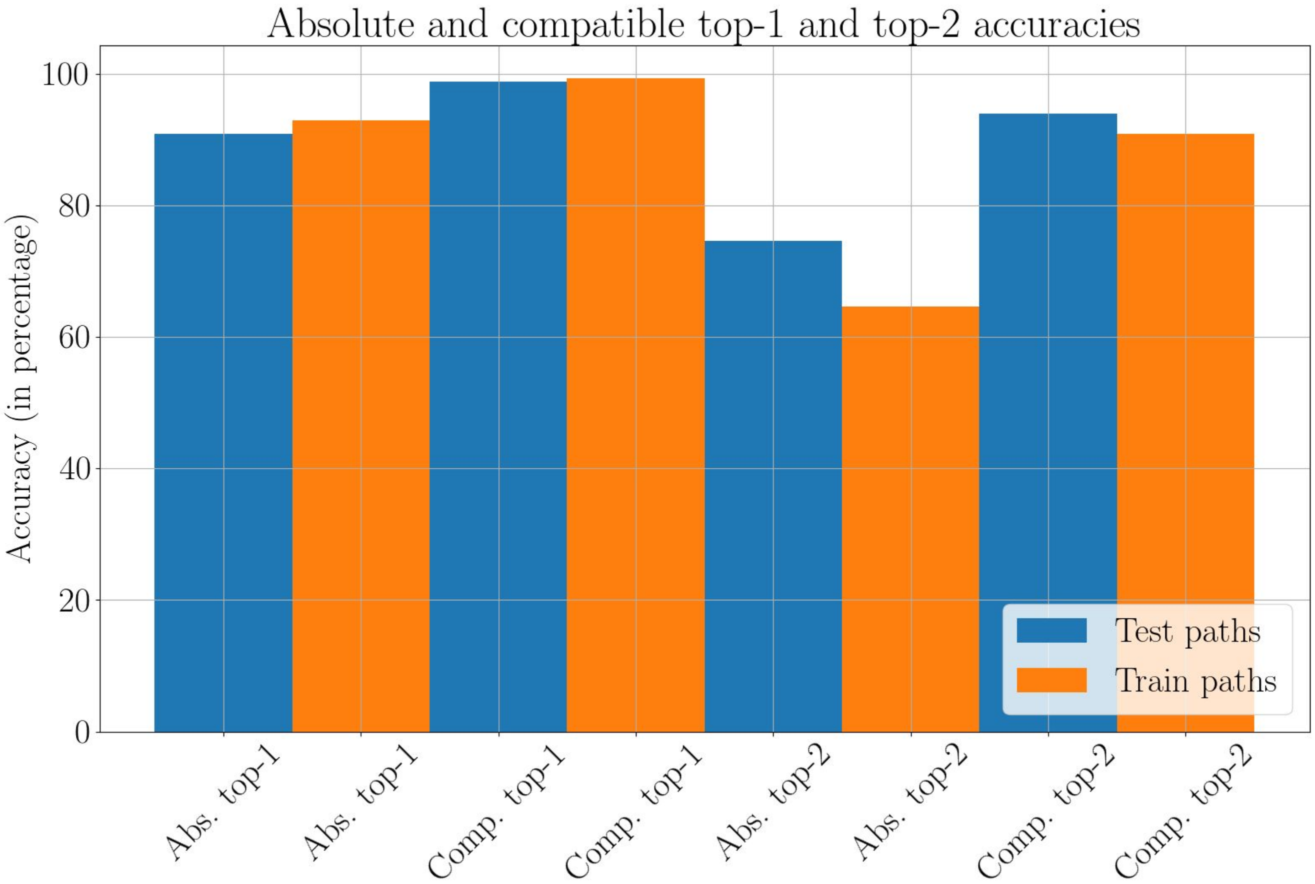
Precision and loss for various dropout values
Absolute precision (a)
Compatible precision (b)
Training loss (c)
Validation loss (d)
Precision
Cross-entropy loss
Training epochs
Training epochs
0.0
0.1
0.2
0.3
0.4



Precision and loss for various number of memory units
Absolute precision (a)
Compatible precision (b)
Training loss (c)
Validation loss (d)
Precision
Cross-entropy loss
Training epochs
Training epochs
64
128
256
512
0.8
0.6
0.9
0.8
0.02
0.01
0.00
0.003
0.002
0.001
0
10
20
30
40

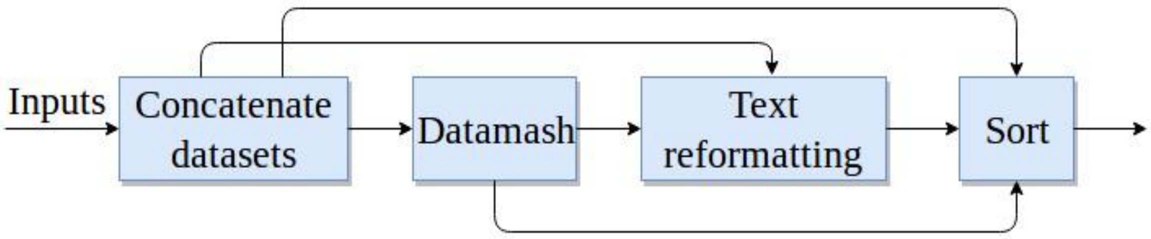


Variation of precision with the length of paths
Precision
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
5
10
15
20
25
Length of paths
Absolute precision (test paths)
Compatible precision (test paths)
Absolute precision (train paths)
Compatible precision (train paths)


Absolute and compatible top-1 and top-2 accuracies
Accuracy (in percentage)
0
20
40
60
80
100
Test paths
Train paths
Abs. top-1
Abs. top-1
Comp. top-1
Comp. top-1
Abs. top-2
Abs. top-2
Comp. top-2
Comp. top-2

Higher order dependency

- Not dependent only on immediate parent
- Dependent on all previous tools



Higher order dependency: https://arxiv.org/pdf/1506.06268.pdf
https://arxiv.org/pdf/1508.03113.pdf

Bayesian network

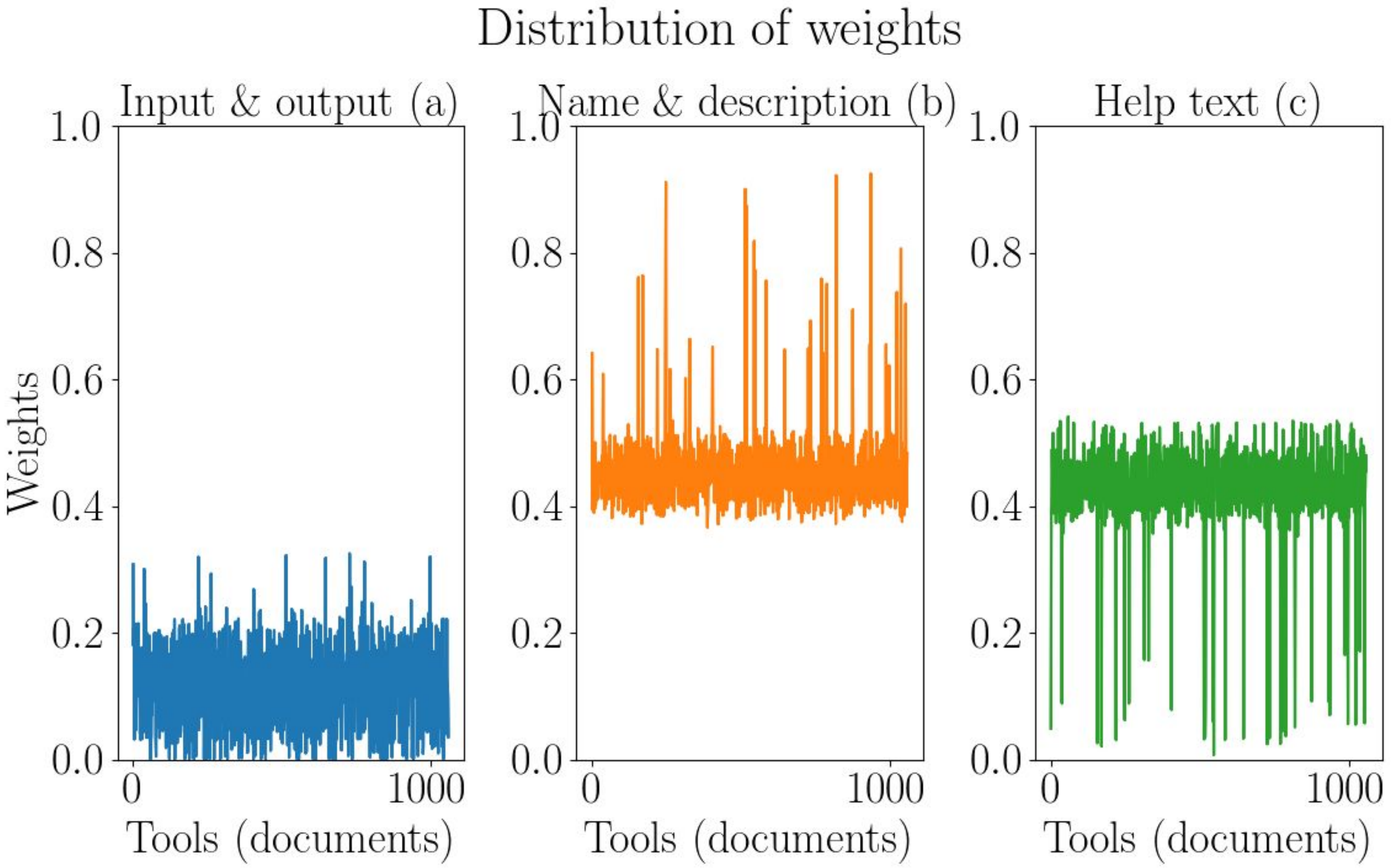
- Explain workflows
- Predict missing nodes
- Compute joint and conditional probabilities
- High computational cost with large number of nodes
- Prediction by probabilistic network is a hard problem

https://www.microsoft.com/en-us/research/uploads/prod/2004/01/Large-Sample-Learning-of-Bayesian-Networks-is-NP-Hard.pdf
https://www.sciencedirect.com/science/article/pii/000437029090060D

Hidden markov models

- Transition and prior probabilities
- Transition matrix is large
- Current state depends only on previous state (first-order)
- Higher number of parameters
- Higher order markov models

https://arxiv.org/pdf/q-bio/0505002.pdf
https://pdfs.semanticscholar.org/8463/dfee2b46fa813069029149e8e80cec95659f.pdf
http://mlg.eng.cam.ac.uk/zoubin/papers/ijprai.pdf


Distribution of weights
Input & output (a)
Name & description (b)
Help text (c)
Weights
1.0
0.8
0.6
0.4
0.2
0.0
0
1000
Tools (documents)