Application of CNNs in MHT problems

Conejo Villalobos César

Department of Statistics
Universidad Carlos III de Madrid

September 21, 2021

Overview

1. Multiple Hypothesis testing (MHT)
2. Convolutional Neural Networks (CNNs)
3. Example 1: Female mice diet
4. Example 2: Normal population
5. Conclusions
6. Appendix

MHT: Introduction

- Testing focused on large-scale data.
- Set of $m > 1$ features. We study each hypothesis test individually.
- Sample units (N): Number of units where the measurements are collected.
- Partition of the m hypotheses into two sets:
  - H_0 is true: m_0.
  - H_0 is false: $m_1 = m - m_0$ (interesting).
  - $p_0 = \frac{m_0}{m}$: proportion true null hypothesis. Assumption: $p_0 \geq 0.9$
- Evidence (against H_0) statistical test: p-value.
- Calibrated p-values: Theoretical sampling null distribution is uniform.
- Focus: Difference between two independent groups. t-test.

MHT: Outcomes when testing m hypothesis

	Declared non-significant	Declared significant	Total
H_0 True	$U = m_0 - V$	V	m_0
H_0 False	$T = m_1 - S$	S	$m_1 = m - m_0$
	$m - R$	R	m

Table: 1. Outcomes of testing m hypothesis under a specified significance level α. (Benjamini and Hochberg, 1995)

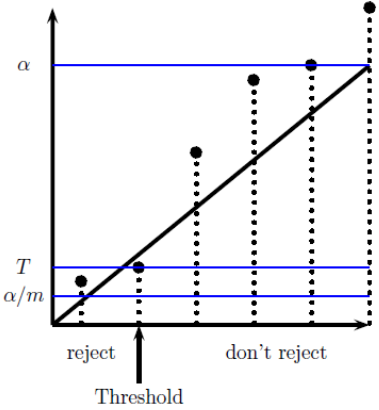
- $R = R_\alpha$: Total number of hypothesis rejected.
- V: Total type-I errors. (FP, False Discovery)
- S: Total TP (True discoveries)
- In practice: m is known, R is an observable RV and V, S are unobservable RVs.

MHT: Classical Procedures

Classical Procedures (CP)

- CP define and control a specific error rate.
- Uncorrected Hypothesis Testing (FPR)
  Rejection (R_{UHT}): $p_i < \alpha$.
- Family Wise Error Rate (FWER)
  Procedure: Bonferroni correction
  Rejection (R_{Bonf}): $p_i < \frac{\alpha}{m}$.
- False Discovery Rate (FDR).
  Procedure: Benjamini-Hochberg (BH)
  Rejection (R_{BH}): $p_{(i)} < \frac{i\alpha}{m}$.

Example: Application of the previous procedures with six ordered p-values.
$R_{UHT} = 4$; $R_{Bonf} = 0$; $R_{BH} = 2$



MHT: Problems

- Dependency of the p-value: Even a tiny cut-off for the p-value can generate many false positives with a high probability.
- Real interest: $\mathcal{P}(H_0|\text{Data})$.
- Low power: Correction procedures reduce the false positive rate but increase the false negatives considerably.
- Null distribution should be known: Classical procedures depend profoundly on the knowledge of the null distribution.

MHT: Alternatives

Collecting evidence

- Based on [Selke et al, 2001].
- Use calibrated p-values for defining a lower bound Bayes Factor.
- Interpretation: Odds of H_0 to H_1

$$LBBF(p) = \begin{cases} -ep\log(p) & \text{if } p \leq e^{-1} \\ 1 & \text{otherwise.} \end{cases}$$

- Example: $LBBF(p = 0.05) = 0.407$, ie $H_0 : 1$ to $H_1 : 2.5$. Not strong evidence against H_0!

Limitations of classical MHT procedures

- Based on [Mary and Roaquin, 2021].
- Proposal: Semi-supervised approach. User does not known null distribution, but has at hand a sample drawn from the null distribution.
- Where the user can obtain this null train sample?
- Previous experiments, expert criteria, part of the data under test, **simulations**, sampling process.

MHT: Proposal: P-value representation

P-value representation

- Set of ordered LBBFs: $\mathcal{B}^m$
  $\{b_{(i)} = LBBF(p_{(i)}) \mid \forall i = 1, ..., m\}$.

- Quotient of the ordered LBBFs:
  Map: $\Psi : \mathcal{B}^m \to \mathcal{M}^{m \times m}$
  For each feature $i = 1, ..., m$:
  $\Psi(b_{(i)}) = \frac{b_{(i)}}{b_{(j)}} = b_{(i)(j)} \; \forall j = 1, ..., m$.

- Scale 0-1
  $b_{\max} = max\{b_{(i)(j)}\}$
  for $i, j = 1, ..., m$.

- Normalized quotient:
  $\bar{b}_{(i)(j)} = \frac{b_{(i)(j)}}{b_{\max}}$.

- **Matrix of relative evidence among test:**

$$\bar{B} = \begin{bmatrix} \bar{b}_{(1)(1)} & \bar{b}_{(1)(2)} & \cdots & \bar{b}_{(1)(m)} \\ \bar{b}_{(2)(1)} & \bar{b}_{(2)(2)} & \cdots & \bar{b}_{(2)(m)} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{b}_{(m)(1)} & \bar{b}_{(m)(2)} & \cdots & \bar{b}_{(m)(m)} \end{bmatrix}$$

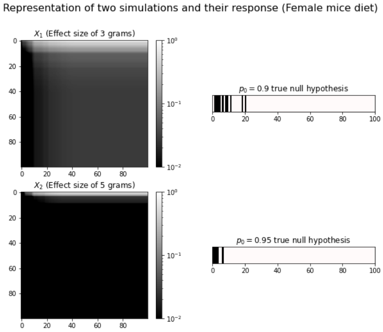
- Input of the supervised algorithm.

MHT: Proposal: Simulation

Supervised Approach

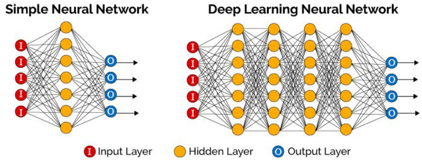
- Multi-label Classification.
- Heavy imbalanced data.
- Performance Metric: Area Under Precision-Recall curve (AUPRC)
- Aggregation: Micro-averaging.
- How to solve the problem? CNNs.
    1. Translation invariance and locality.
    2. Curse of dimensionality.

Example: Matrix representation and response of two independent MHT problems with $m = 100$ features.



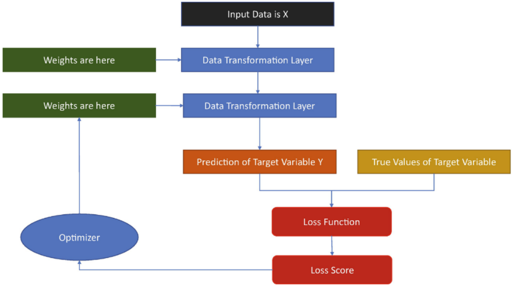
CNNs: Deep Learning

Deep (and breadth)
Number of hidden layers and number of neurons.
Basic architecture: Fully connected neurons



Increase the features $\sim$ increase the complexity of the model

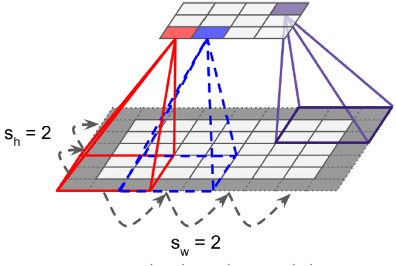
Learning Adjusting the weights via back(for)ward propagation algorithm.



CNNs: Architecture

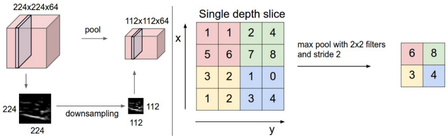
Convolutional Layer

- Spatial convolution over images.
- Parameters:
  1. `kernel_size`
  2. `strides`
  3. `padding`
  4. `filters`



Pooling Layer

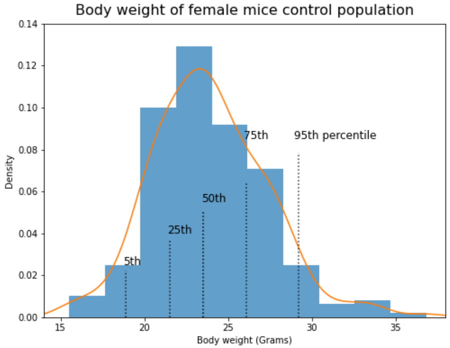
- Downsampling procedure.
- Parameters:
  1. `pool_size`
  2. `strides`
  3. `padding`
  4. Type of aggregation: max, mean.



Simulation 1: Female mice diet

	Value
Population (Individuals)	225
Mean	23.89 (g)
Standard deviation	0.22 (g)
Minimum	15.51 (g)
Maximum	38.84 (g)

Table: 2. Statistical summary female mice body weight.



Simulation 1: Female mice diet $m = 100$.

- Number of features: $m = 100$,
- Sample units: $N = 12$,
- Effect size: 96 distinct values, from 0.5 to 10.0 g, with a difference of 0.1 grams.
- Proportion true H_0, 10 distinct values, from 0.9 to 0.99 (Difference of 0.1 pp)
- MHT problem:
  - H_{0i}: Diet i does not affect weight.
  - H_{1i}: Diet i is effective.

ANN and CNN architectures

(a) *ANN architecture*

Model: "sequential"

Layer (type)	Output Shape	Param #
flatten (Flatten)	(None, 10000)	0
dense (Dense)	(None, 500)	5000500
dense_1 (Dense)	(None, 250)	125250
dropout (Dropout)	(None, 250)	0
dense_2 (Dense)	(None, 100)	25100

Total params: 5,150,850
Trainable params: 5,150,850
Non-trainable params: 0

(b) *CNN architecture*

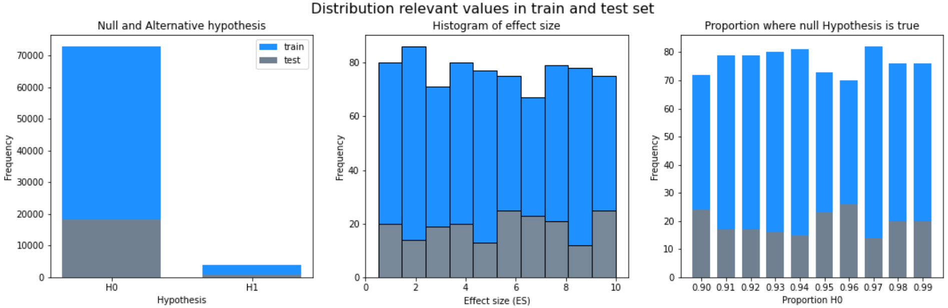
Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 100, 100, 2)	20
max_pooling2d (MaxPooling2D)	(None, 50, 50, 2)	0
flatten (Flatten)	(None, 5000)	0
dense (Dense)	(None, 500)	2500500
dense_1 (Dense)	(None, 250)	125250
dropout (Dropout)	(None, 250)	0
dense_2 (Dense)	(None, 100)	25100

Total params: 2,650,870
Trainable params: 2,650,870
Non-trainable params: 0

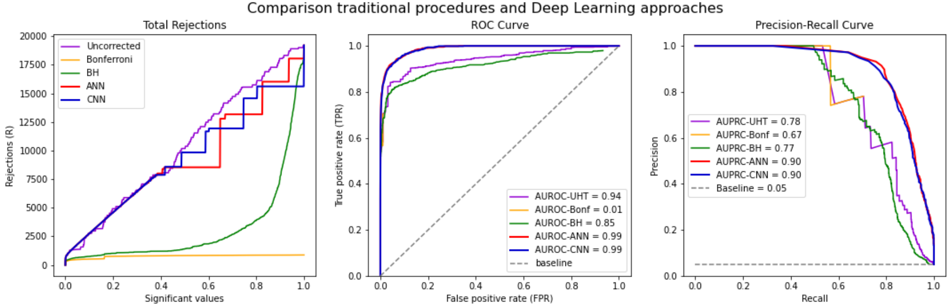
Simulation 1: Split train and test

- Total MHT scenarios: 960 (70% train, 10% validation, 20% test).
- Total features: $960 \cdot 100 = 96\,000$.
  - True H_0: 91 104 diets (72 876 training and 18 228 testing)
  - True H_1: 4 896 diets (3 924 training and 972 testing)



Simulation 1: Performance in test set

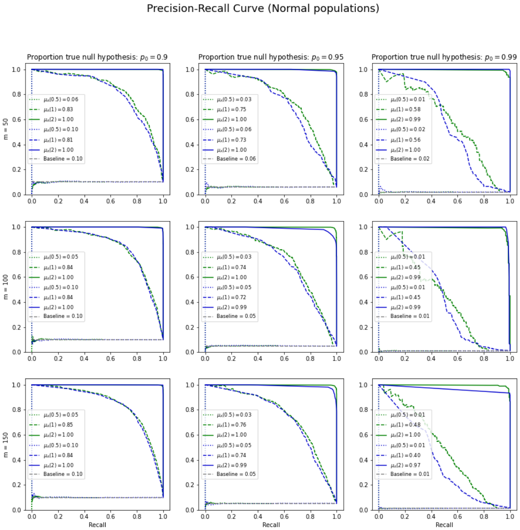
- Total features test set: 19 200 (H_0:18 228, H_1:972)
- AUPRC of CNN (and ANN) 0.9. (Both train and test)
- DL: Precision and recall greater than 80% simultaneously in test set.



Simulation 2: Normal population

- $X_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, for $i = 1, ..., m$ independent populations and unknown variance σ_i^2.
- MHT problem: $\{H_{0i} : \mu_i = 0 \text{ versus } \mu_i \neq 0 \ \forall \sigma_i^2 > 0\}, i = 1, ..., m$
- Simulation parameters:
  1. Total simulations: $B = 100$
  2. $N = 20$,
  3. $m = 50, 100$ and 150,
  4. $p_0 = 0.9, 0.95$ and 0.99,
  5. $\mu_i = \mu_A$ where $\mu_A = 0.5, 1$ and 2. and $\sigma_i^2 = 1$
  6. Benchmark: BH procedure.

Simulation 2: Normal Population



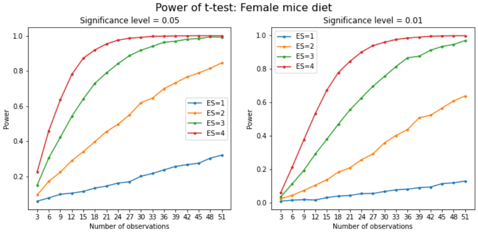
Conclusions

- CNNs offers a competitive alternative of detecting significant features in the context of MHT problems.
- Simulation case 1 (female mice diet) gives satisfactory results for detecting a higher number of cases where the alternative hypothesis is true.
- Simulation 2, shows that we can not potentially generalize a predefined architecture to another datasets or different number of features.
- Future analysis:
    1. Not calibrated p-values: Empirical null distribution of the p-values estimated directly from the data
    2. Curse of dimensionality: Explore sparse representation

Q & A

Appendix 1.1: Hypothesis Testing

- Reliability: Main factors:
  1. Significance level: It is the standard of proof that the phenomenon exists or the risk of mistakenly rejecting the null hypothesis. Implies directly the critical region of rejection.
  2. Power: Probability of rejecting the null when the null is false
  3. Effect size: Degree to which the phenomenon is present in the population.
  4. Sample size: Number of observed samples from the population.



Appendix 1.1: P-values

- Distance between the data and the model prediction is measured using a test statistic.
- P-value is the probability that the chosen test statistic (t-test) would have been at least as large as its observed value if every model assumption were correct.
- p-values are no longer useful quantity to interpret when dealing with high-dimensional data (Many FP with high probability, increasing the features increases the error just by chance)
- Most common MHT methods are based on the evidence provided by test statistics and their corresponding p-values.
- Other ways of collecting evidence? Bayes Factors (BFs).
- BF: likelihood ratio of the alternative against the null hypothesis.
- However, fully defined and interpretable BFs require heavy computational techniques for being adjusted.
- Goal: Interpret the calibrated p-values as lower bounds on Bayes Factors.

Appendix 1.1: Calibration of p-values: LBBF

- Under appropriate test statistic T, larger values would be evidence in favor of H_1.
- Density of p under H_1 should be decreasing in p.
- Consider alternative distributions for the p-values. Selke et al. procedure:

$$H_0 : p \sim \text{Uniform}(0,1) \text{ versus } H_1 : p \sim \text{Beta}(\xi, 1) = \xi p^{\xi-1}$$

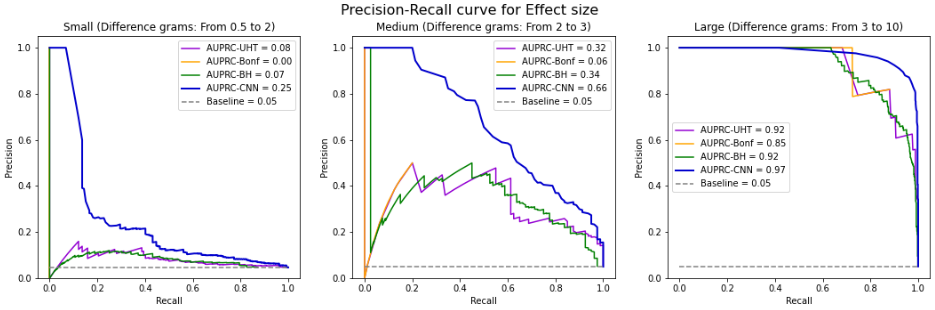
- Why Beta$(\xi, 1)$? Beta is easy to work! With $\xi = 1$, we have H_0.
- Remember: posterior odds $=$ prior odds $\times$ Bayes Factor.
- BF (or odds) of H_0 to H_1 for a given prior density $\pi(\xi)$ is:

$$\text{BF}(p) = B_\pi(p) = \frac{p}{\int_0^1 \xi p^{\xi-1} \pi(\xi)\, d\xi}$$

- $\text{LBBF}(p) = \inf B_\pi(p) = \frac{p}{sup_\xi \xi p^{\xi-1}} = -eplog(p)$ for $p < e^{-1}$.

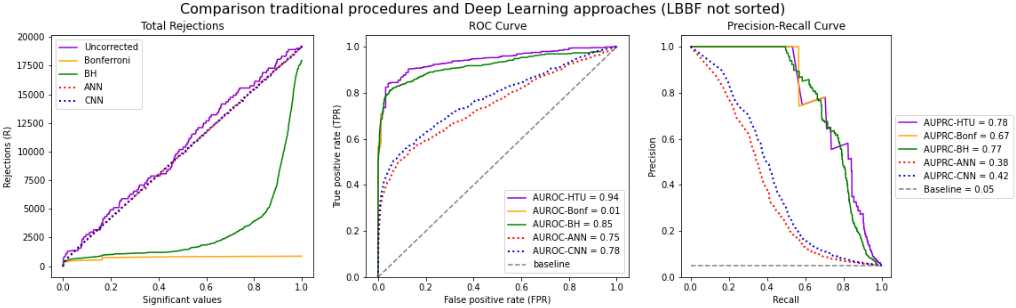
Appendix 2.1: Simulation 1: Performance for effect size

- Effect size depends on the unit of measurement (grams).
- Universal effect size index: Conhen's d. $d = \frac{|m_A - m_B|}{s}$.
- Small ($d = 0.2$), medium ($d = 0.5$), large $d = 0.8$.
- Based on $B = 2000$ simulations, we compute the median ES index for the difference in weight ranging from 0.5 to 10 grams.



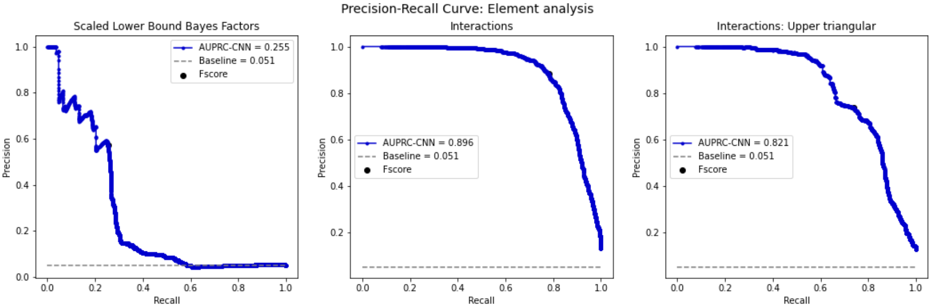
Appendix 2.2: Simulation 1: Order LBBF

- Primary assumption CNN: Compositional data (translation invariance and feature locality)
- In case that we assign randomly position of the LBBF, there is a considerable reduction of the performance in the test set. Overfitting!



Appendix 2.3: Simulation 1: Interaction LBBF and Sparse models

- AUPRC-CNN test set using the complete representation: 0.9
- If we consider only the scaled LBBF, we have AUPRC-CNN = 0.26
- LBBF interaction is crucial in the performance of the models. AUPRC-CNN = 0.89
- If we consider only the upper part matrix of the representation, AUPRC-CNN = 0.82.



Appendix 3.1: Simulation 3: Curse of dimensionality

- ANN and CNN architectures for m = 1000 features
- ANN: 1500 neurons, 300 million parameters
- CNN: Adding several convolutional/max-pooling layers reduces the total parameters around 3 million.

(a) *ANN architecture*

Model: "sequential"

Layer (type)	Output Shape	Param #
flatten (Flatten)	(None, 1000000)	0
dense (Dense)	(None, 300)	300000300
dense_1 (Dense)	(None, 250)	75250
dropout (Dropout)	(None, 250)	0
dense_2 (Dense)	(None, 1000)	251000

Total params: 300,326,550
Trainable params: 300,326,550
Non-trainable params: 0

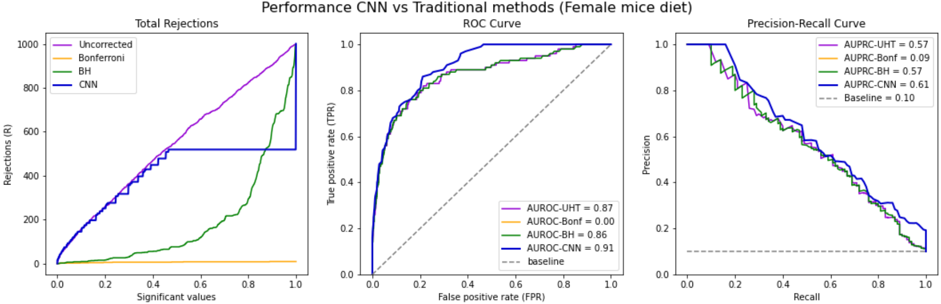
(b) *CNN architecture*

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 1000, 1000, 2)	20
max_pooling2d (MaxPooling2D)	(None, 500, 500, 2)	0
conv2d_1 (Conv2D)	(None, 500, 500, 4)	76
max_pooling2d_1 (MaxPooling2	(None, 250, 250, 4)	0
conv2d_2 (Conv2D)	(None, 250, 250, 8)	296
max_pooling2d_2 (MaxPooling2	(None, 125, 125, 8)	0
conv2d_3 (Conv2D)	(None, 125, 125, 16)	1168
max_pooling2d_3 (MaxPooling2	(None, 63, 63, 16)	0
conv2d_4 (Conv2D)	(None, 63, 63, 32)	4640
conv2d_5 (Conv2D)	(None, 63, 63, 32)	9248
max_pooling2d_4 (MaxPooling2	(None, 32, 32, 32)	0
conv2d_6 (Conv2D)	(None, 32, 32, 32)	9248
max_pooling2d_5 (MaxPooling2	(None, 16, 16, 32)	0
flatten (Flatten)	(None, 8192)	0
dense (Dense)	(None, 300)	2457900
dense_1 (Dense)	(None, 250)	75250
dropout (Dropout)	(None, 250)	0
dense_2 (Dense)	(None, 1000)	251000

Total params: 2,808,846
Trainable params: 2,808,846
Non-trainable params: 0

Appendix 3.2: Simulation 3: Female mice with m = 1000 features

- $N = 12$, ES ranging from 2 to 15 grams, (0.1 grams distance) and $p_0 = 0.9, 0.95$ and 0.99.
- Total simulations: 390 (281 for training)
- Figure below is used only for a single prediction. Still, we have a competitive model.



Appendix 4.1: Simulation 1: CNN architecture

- First layer: Convolutional.
  1. Number of filters: 2, Kernel size: 3×3, stride: 1.
  2. Padding: **same** (zero-padding)
  3. activation: **ReLu**
  4. Kernel Initializer: **HeUniform**. (Remember, p-values are calibrated).
- Second layer: Max pooling
  1. Pool size: 2×2, stride: 2×2, padding: **same** (Reduction to the dimensions to the half)
- Hidden layers: Fully conneted layers:
  1. Third layer: Fully connected: 500 neurons, activation: **ReLu**, kernel initializer: **HeUniform**
  2. Fourth layer: Fully connected: 250 neurons, activation: **ReLu**, kernel initializer: **HeUniform**
  3. Fifth layer: Dropout: 0.5 probability
  4. Sixth layer: Putput layer: 100 neurons, activation: **sigmoid**
- Loss function: **binary_crossentropy**
- Optimizer: Nadam() (Default settings, high convergence speed and quality)
- Metric: AUPRC

References

David Mary and Etienne Roaquin (2021)
Semi-supervised Multiple Testing
*arXiv*2106.13501.

Selke, T., Bayarri, M., and Berger, J.O. (2001)
Calibration of p values for testing precise null hypothessis
*The American Statistician*55(1), 62 – 71.

The End