Semi-supervised learning for nucleic acid cross-linking mass spectrometry

Emil Paulitz

July 21, 2020

Contents

1 Introduction

- Motivation

2 Background

- Proteomic, MS, Peptide Identification, Protein-RNA cross-linking (alles relativ kurz)

Nachweise! Rather straightforward algorithms like (Beispiele einfacher scoring algorithmen) compare the spectra with every possible peptide and compute

a score based on their similarity. The best scoring peptide is then considered a peptide-spectrum-match (PSM). Those algorithms often differ greatly in their results and are thus not considered reliable, but their scores enable FDR estimation using decoy databases and serve as a basis for score re-calibration with the Percolator algorithm.
Decoy databases are created from the target database, have the same size and contain all of the proteins or peptides in a usually reversed or shuffled order. They are presented to the scoring algorithm along with the target database. For the algorithm, decoy and target peptide are not distinguishable, and thus, for spectra not matching any target peptide very well, the best scoring peptide will be a decoy half the time. This allows for an estimation of wrongly assigned targets, since the score distribution should be the same for decoys and false targets.
In practice, one estimates the probability of a PSM being a false target via counting the number of decoy-PSMs with the same or a higher score. It is then assumed, there are as many false targets and thus a false discovery rate (FDR) can be calculated. This leads to the following formula:

$$FDR = \frac{\#\text{ false target PSMs}}{\#\text{ all target PSMs}} = \frac{\#\text{ decoy PSMs}}{\#\text{ all target PSMs}} \tag{1}$$

Currently, this formula, yielding slightly lower values for the relevant, low FDRs is used [Nachweis, (percolator paper?)]:

$$FDR = \frac{\#\text{ decoy PSMs}}{\#\text{ all PSMs}} = \frac{\#\text{ decoy PSMs}}{\#\text{ decoy PSMs} + \#\text{ target PSMs}} \tag{2}$$

The q-value for a PSM is then derived from this as the minimum FDR of all PSMs with a lower score. (q-val Definition und Grund für diese Berechnung nachschauen)
- Score Rekalibrierung mit Percolator

3 Material, Methods

- Material: Was ich für ein Datensatz zum Testen benutzt habe und wo der herkommt

3.1 Implementation of the percolator algorithm

- Wie genau stelle ich das vor, siehe Mail? Wichtige Punkte wären:
- Verwendete Scipy-Methoden
- Abbruch wenn es nicht besser wird und dass ich die AUC als Metrik nutze
- feature normalization
- Wichtige Hilfsfunktionen (pseudoROC zB)

3.2 Improvements of the percolator algorithm for cross-link identification

- Klassenspezifische q-values
- ROC nach jeder Iteration und aufgesplittet nach XL/nXL

3.2.1 How to deal with different Ranks

- OptimalRanking Option (Erst paar Iterationen Ränge verändern lassen und dann die schlechten entfernen)

3.2.2 Characteristics of cross-linking PSM datasets

- Verhältnis Targets:Decoys und XL:non-XL in inneren und äußeren splits gleich lassen und MinMaxMedian Auswertungen mithilfe von google colab cloud computing
- Imputation (kam zwar nichts raus ist aber trotzdem interessant)
- Trennung von Datensatz nach XL/nXL oder sogar cross-linking target falls Datensatz groß genug

3.2.3 Small datasets

- Ratio Testing (nicht-random aus ganzem Datensatz und random aus Top 10%. Liefert Erkenntnisse über die mögliche Größe des Datensatzes und eventuell die Sinnhaftigkeit, wann man die Datensätze einfach trennen kann → Für den Leser relevant)
- Einbau von Identifikationen bei 1% FDR als Metrik (Sinnhaftigkeit kann man ja diskutieren)

(- Performance auf anderem Datensatz
- Vergleich mit Entrapment FDR)

4 Results

4.1 Implementation of the percolator algorithm

- Reimplementierung funktioniert wie Original
- feature normalization war wichtiger boost
- ROC nach jeder Iteration zeigen

4.2 Improvements of the percolator algorithm for cross-link identification

4.2.1 How to deal with different Ranks

- Ergebnisse von OptimalRanking

4.2.2 Characteristics of cross-linking PSM datasets

- Verhältnis Targets:Decoys und XL:non-XL verringt die Streuung: MinMax-Median Auswertungen
- Bei Imputation kam nichts heraus
- Großer Unterschied wenn man den (großen) Datensatz nach XL/nXL oder sogar cross-linking target aufteilt

4.2.3 Small datasets

- Sinnvolle Plots zu Ratio Testing
- Neue Metrik erlaubt es der Implementierung, auch auf kleineren Datensätzen zu funktionieren

5 Discussion

- Methoden hinterfragen oder begründen, Ergebnisse interpretieren, Anwendbarkeit diskutieren, z.B.:
- Warum habe ich mich mit Rängen beschäftigt? (Bei XL-Datensätzen oft sinnvoll um erst percolator entscheiden zu lassen was auf Rang 1 steht (?)) - C Parameter für jeden split neu optimieren führt zu overfitting? → Original-Algorithmus macht es auch so
- Wie sinnvoll ist die neue Metrik? - ScanNr Versuche: Gleiche Spektren (identifiziert anhand der ScanNr.) auf verschiedene splits verteilen verändert nichts, d.h. vermutlich sind die niedrigeren Ränge dann so schlecht, dass es nichts bringt die schonmal gesehen zu haben.
- Peptide Versuche: Schlechtere Ergebnisse, aber vllt ehrlicher?
- Mögliche weiterführende Experimente: mächtigere Klassifikatoren + monotonic constraints (wie von Timo ausprobiert), Ada-Boosting, feature selection