Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Manuscript Category

OXFORD

Protein Classification

Predicting Protein Subcellular Localization from Sequence Data

Eric Hambro [1,*]

[1] Department, Institution, City, Post Code, Country

* To whom correspondence should be addressed.

Abstract

Motivation: Deriving protein function from sequence level data can provide. Using features derived from the sequence and a random forests model was trained to classify protein sequences according to their location of use. Using a validation set of blah, an accuracy prediction accuracy of blah was reported, along with an F1 score of blah.
Results: Lorem ipsum Text Text Text Text Text Text Text Text Text Text Text
Availability: Text Text Text Text Text Text
Contact: eric.hambro.17@ucl.ac.uk

1 Introduction

Although sequence data of genes has proliferated since the advent of **technique**, the classification of protein function has lagged behind significantly. **Statistic**. This is partly down to the fact that protein function is a complex labelling task, given the function of proteins may vary under different conditions. In cases where no homologues are present for a gene, knowing the subcellular location of the the protein gives indicator as to the proteins function. Furthermore discovering subcellular location has proven to be vital information in the research of new drugs and vaccinations. As such, automated means of predicting subcelular localization are utmost urgency in todays drug search enbironment.

Since as early as **blah**, machine learning has been used to predict automatically the subcellular localization of a protein, in an attempt to reduce the cost and time taken to process the proliferating protein sequences. Traditional methods such as **experiment**, which were laborious and time intensive, have been replaced in parts by papers, which are able to make predictions at a high degree of confidence *reference* with only sequence data.

Some these techniques include **machine learning technique**, which does a **xyz**, and **ml technique 2** which does **this**.

In this particular experiment we are presented with a training data set X proteins, each corresponding to one of four subcellular locations, and we train and evaluated a Random Forests classifier on these data, finally making predictions of locations for X unknown proteins.

Section 2 of this report contains a description of the method used, and Section 3 presents our results. A discussion of these results is then presented in Section 4, along with analyses of the features of the model, a comparison to another model, and our motivation in choosing this method. Finally, I present a comparison with existing work in the field and further work that could be done, ending with a conclusion.

2 Background

2.1 Subcellular Localization & Transport

In eukaryotic cells, proteins are often highly adapted to fulfilling their specialist functions in the cell. These functions can be very localized within organelles or other parts of the cell, and so the proteins are too adapted to the specific subcellular location in the cell. In order to faciliate transport of protein to its location, often a location signifier is in some way encoded into the structure of the protein. This what's known as a target peptide (if heading to an organelle) or signal peptide (if secreted). These peptides can vary greatly in structure, depending on the final destination, and can be explored in greater depth in Alberts, *et al*. (2008). Sometimes they are removed by a signal peptidase, when they reach their destinations. Below are some of the known features of signal and target peptides, as mentioned in the above book.

1. **Targeting Secretion**
   Proteins headed toward secretory pathways often contain short amino acid encodings (say 16-30 amino acids long), occuring near the N-terminus of the protein. Often they include a domain or sequence that is particularly hydrophobic, and these hydrophobic residues may form an α-helix.
2. **Targeting Mitochondria**
   In the case of proteins headed to mitochondria, a *mitochondrial targeting signal* of a peptide signal is often found at the N terminus of the protein. These often consist of alternating positively charged and hydrophobic amino acids, that come together to form an amphiphilic

α-helix. These this pattern (of positively charged residues on one side, and uncharged hydrophobic residues on another) is what is primarily recognised by the receptor proteins, rather than a specific sequence.

3. **Into The Nucleus**

   Proteins that require importing into the nucleus often contain a *Nuclear Localization Signal*, which may consist of a sequence amino acids that are positively charged (often arginine and lysine) exposed on the surface of the protein. In contrast with the other target peptides, it appears these signals can occure almost anywhere in the chain, and often form loops. Many of these are still uncharacterised.

4. **Out of the Nucleus (Into The Cytosol)**

   Proteins that are are being exported out of the nucleus and into the cytosol are often tagged with a *Nuclear Export Signal*. A known example of *Nuclear Export Signal* consists of a chain of several (hydrophobic) leucines, interspersed with other amino acids, between them. Since some proteins frequently move between the nucleus and the cytosol, it is common for some proteins to contain both *Nuclear Export Signals* and *Nuclear Localization Signals*, which are effectively controlled by the cell to direct movement.

2.2 Bagging & Boosting

Our investigation compares two machine learning models to a benchmark random classifier, selecting the best. These two models (Random Forests and AdaBoost) were selected on account of being good are examples of common ensembling techniques: *bagging* and *boosting*.

2.2.1 Bagging & Random Forests

Bootstrap aggregation (or *bagging*) is an ensemble method to the reduce the variance of a prediction when using high variance classifiers. It involves, selecting many subsamples of the training data, and generating estimates from the average performance of classifiers trained on this data. Thanks to this averaging of predictions, overfitting our training data becomes less of a risk, since each classifier can overfit to it's subsection of the data without impacting the average too greatly in the end.

In a Random Forest, our high-variance classifier is a decision tree, and we adapt the *bagging* algorithm to improve performance. Specifically we only give each individual classifier access to a random subset of the features of the data, at training time. The aim of this is to stop any correlation on the decision trees that may arise from their greedy behaviour, as they look through variables to select the best split point. By only having access to a random subset of features, the decision trees become less correlated, and the Random Forest algorithm produces a better 'averaged' estimate.

2.2.2 Boosting & AdaBoost

The AdaBoost algorithm is a different method that uses a ensemble technique known as *boosting*. Instead of a combining votes from many high variance normal classifiers (like a decision tree), *boosting* involves learning by combining lots of weak learners (like a decision stump), by training the on particularly chosen subsections of the data, and combining votes in a particular way. Specifically, each successive weak-learner is trained on selections of the data that aim to emphasise the most misclassified data points. These learners are then weighted by their error, to provide the overall classification.

3 Method

3.1 Preprocessing

Data was obtained from four files containing protein sequences in Fasta format. Each file contained proteins according to its type of subcellular localization: cytosolic, secreted, nuclear and mitochondrial. The number of proteins in each file is presented in Table 1. These data files contained no homologues, and therefore could be assimilated, shuffled and split into a training and validation set randomly. The data was then split into a test set that would be held out for final evalution (10%), and a train and validation set (90%) that could be used to tune hyperparameters.

Table 1. Sample Data Set By Protein Class

Cytosolic	Nucleic	Secreted	Mitochondrial
row4	row4	row4	row4

The data was then fed through a pipeline to extract a number of features from the sequence. These features all corresponded to floating point numbers, and so were concatenated to form a 75-dimensional feature vector, that would be used to train the models.

3.2 Features

The open source 'BioPython' *reference* module was used to extract features from the sequence data, allowing us to augment raw residue sequence data with important characteristics of the amino acids, such as isoelectronic point or molecular weight. In total 75 features were created, although many of these could be grouped into the same category. These features were:

- **Protein Length** - An integer representing the length of the protein sequence, in residues.
- **Amino Acid Composition** - A float created for each amino acid, corresponding to the percentage composition of that amino amino acid in the protein.
- **Aromaticity** - A float corresponding to a measure of the aromaticity of the molecule according to Lobry et al*1994*. This is measured by adding the frequencies of residues phenylalanine, tryptophan and tyrosine, in the sequence of the protein.
- **Isoelectric Point** - A float representing the isoelectric point of the protein according to values taken by Bjellqvist et al *ref*.
- **Molecular Weight** - A float simply molecular weight of the protein.
- **Secondary Structure Features** - Three separate features were created indicating the fraction of amino acids which tend to be either α-helixes, β-sheets, or turns/loops. These amino acids were: α - Val, Ile, Tyr, Phe, Trp, Leu; β - Glu, Met, Ala, Leu; *loops* - Asn, Pro, Gly, Ser
- **Start/End Amino Acid Composition** - Two floats were created for each amino acid, according to the percentage composition of that amino amino acid in the first and last 20 residues of the protein respectively.

These features were chosen as they had a high impact on the on the remaining the accuracy of the classifier. The importance of each of these features to various classifications is examined in section 4.

3.3 Classifier Training & Tuning

A number of classifiers were trained to investigate the best possible model in predicting the subcellular location. These models were:

- Categorical Distribution
- Random Forests Classifier - One-Vs-Rest (4 Models)
- Adaboost Classifier

All classifiers were implemented using the open source 'scikit-learn' package. In all cases, model parameters were fit with a 5-fold cross validation, on the train and validate set mentioned previously.

Hyperparameters were tuned using a grid search **plot**, and chosing the parameters that had the best 5-fold cross validation score of accuracy.

In the Random Forests Model, multiple models were trained, each solely to predict one class. Although this is strictly unnecesssary for a Random Forests Classifier, it provides the additional advantage of allowing for interpretable feature importances, and more interpretable analysis of the individual clusters. A single multi class Random Forests Model was also implemented, though this was found to have the same accuracy as the individual models, to within experimental error due to the stochasticity of the models.

In this multi-model case, 'scikit-learn' provided the best means for creating the 4 mutually exclusive classifiers from Random Forests, using it's OneVsRestClassifier.

3.4 Benchmarks

A raw, bottom level benchmark for this investigation was a random classifier. This classifer was simply an implementation of a categorical distribution, with the probability vector learnt from the frequency of each category in the training dataset.

$$x \sim Categorical(\mathbf{p}) \text{ where } \mathbf{p} = [0.25, 0.25, 0.25, 0.25]$$

4 Results

4.1 Final Hyperparameter

For each model, grid search found the following hyperparameters provided the best test cross validation score.

table hyper params

Table 2. Hyperparameters of Model

Model	Hyperparameter	Settings
Categorical (Benchmark)	N.A.	N.A.
Random Forests Classifier (OvM)	estimators	450
Adaboost Classifier	estimators	200
Adaboost Classifier	learning rate	0.25

4.2 Model Scores

All models were evaluated on final held-out test set. Calculations were made of the accuracy, the F1 score and Area Under Curve of the Returning Operator Characteristic (AUCROC)

Table 3. Hyperparameters of Models

Model	Cross Val Acc	Test Acc	F1 Score
Categorical (Benchmark)	x	Test Acc	row1
Random Forests Classifier	z	Test Acc	row3
Adaboost Classifier	w	row4	row1

4.3 Predictions

Given the above evaluation of the models, a Random Forest Model was selected as our best model, on account of its X score. As as result we used it to make predictions of **YYY** hitherto unclassified proteins. These prediction, along with per centage confidence scores, are given in table *blah*.

Table 4. Predictions of Test Proteins

Protein ID	Subcellular Location	Probabilty
row1	row1	row1
row2	row2	row2
row3	row3	row3
row4	row4	row4

This is a footnote

4.4 Plots

Finally, to investigate the importance of various features, the ROC curve was plotted for the best model, Random Forests, along with the confusion matrix, and the importance weights for the random forests classifier. These are presented *here*.

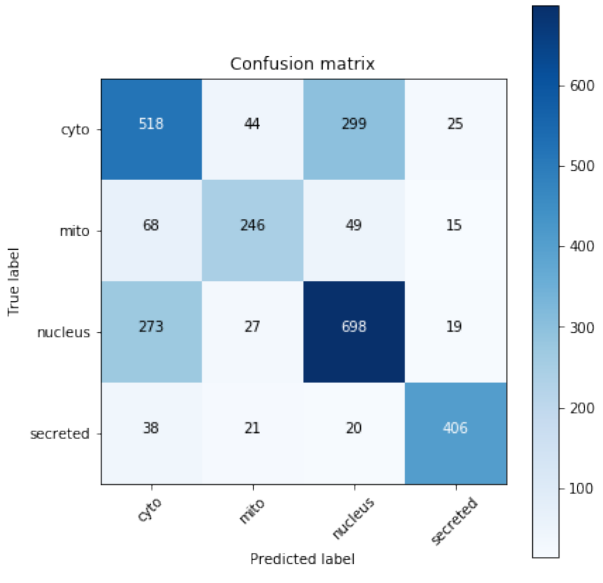
5 Discussion

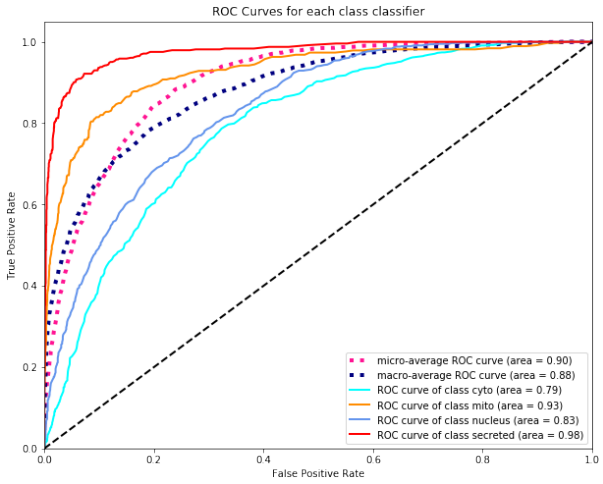
5.1 Classifier Comparison

From the above results it is quite clear that both boosting and bagging methods far outperformed the benchmark method. The woeful performance of X accuracy is to be expected for a model that doesn't take into account any of the sequence data, or underlying features.

It is also clear that Random Forests outperforms AdaBoost, securing a decisive performance improvement of X compared to Y. In both cases this *who got more false positives, false negatives*.

This performance difference is likely down to the difficulty that *boosting* methods have in dealing with high variance datasets..





5.2 Error Analysis

In inspecting our best performing model, Random Forests, more closely, it is clear that the errors in the model were not distributed evenly across categories. Instead it appears that the greatest source error in fact come from misclassifying nuclear labels for cytosolic ones. This can be seen clearly from the confusion matrix presented in *confusion matrix*. Almost $X\%$ of nuclear proteins are misclassified as cytosolic, and $Y\%$ as cytosolic as nuclear.

This misclassification is made all the more evident by looking at the Returning Order Characteristic (ROC) curves, where all the classifiers are plotted together. From these graphs, it is immediately clear that the best classifier is the secreted class, with the area under curve (AUC) of 0.98, followed by the mitochondrial class of 0.93. The worst performer is the cytosolic class (0.79) followed by the nuclear class (0.83).

There are a number of reasons why it is conceivable that these particular protein classifiers (cytosolic and nuclear) should perform worse than the other two categories (secreted and mitochondrial). For one thing, there are a number of proteins that frequently move between the nucleus and the cytosol, meaning that they may posses both *Nuclear Export Signals* and *Nuclear Localization Signals*. This mixup of very distinct features (in the form of target peptides) would likely lead the classifiers to struggle to distinguish the correct class in ambiguous cases.

Furthermore, it is likely that mitochondrial proteins and secreted proteins may have important properties that are unrelated to target peptides but still show up in our choice of features. For instance, since secreted proteins need to cross the cell membrane, it is possible features such as very large size or certain peptide make up, might make this action difficult. There may exist alternative restrictions on mitochondrial proteins, that need to be adapted to help in respiration, which might also restrict the protein's structure.

At any rate it is clear from the ROC curves in particular that secretion is particularly well detected, with *comment about false positives etc*, nuclear and cytoplasmic classes are easier to confuse. An analysis of the exact features that could be responsible for this are presented in the next section.

5.3 Features Analysis

Perhaps more interesting than the comparison of the two different models is the close comparison of the features that have had a significant impact on the model. Thankfully, our One-Vs-Rest Random Forest Model has an in built way of examining the features, via the feature importances. These are displayed in figure ref importances.

The first thing to notice is the relative scales of the four graphs, in comparison to one another. The secreted classifier has a much stronger set of "significant" features, than all the other classes. In the case of comparison to nucleus and cytoplasmic classes, secreted's top feature is over twice the importance of the latter classes highest feature. Furthermore its top 4 features features all have higher importances than the highest single feature for these classes. This indicates that there are distinct "killer features" for this class, and it appears they are highly tied to the fraction of Leucines and Cysteines at the start of the sequence, and the molecular weight and sequence length. **investigate data?**

Next to notice is that, as mentioned in the initial background, mitochondrial proteins also have importances supporting the target peptides alluded to. *Mitochondrial targeting signals* sometimes consist of alternating positively charged and hydrophobic amino acids forming a helix, so unsurprisingly our strongest features include features noting isoelectricity, structural helices, and the presence of positively charged Arginine at the start of the sequences.

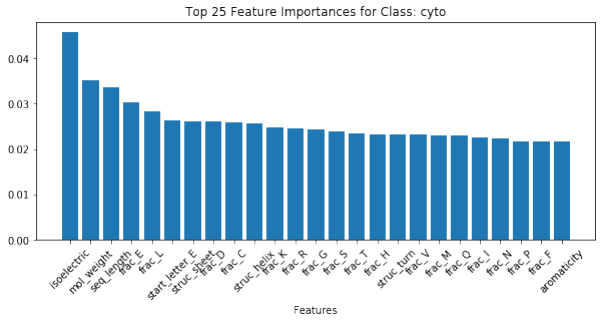
When it comes to nuclear and cytosolic proteins, it's worth noting that the second and third best importance features are shared - indicating the difficulty these classes might have in distinguishing themselves. Furthermore these features - molecular weight and sequence length - are broad macroscopic features that do not really indicate a particular target sequence, and also show why it might be hard to classify these proteins. Even the "strongest feature" for these two classes is a macroscropic feature - isoelectric point, and percentage of "common helix" amino acids, indicating our set of features haven't been able to pick up the nuclear target peptides.

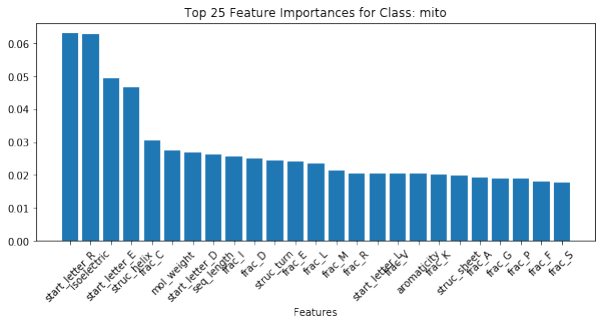
Finally it is worth noting that none of the most important features correspond to peptides at the end of the sequence. This is fairly good evidence that something important happens at the N-terminus of a protein, and supports background theory that this is where target peptides are often found.

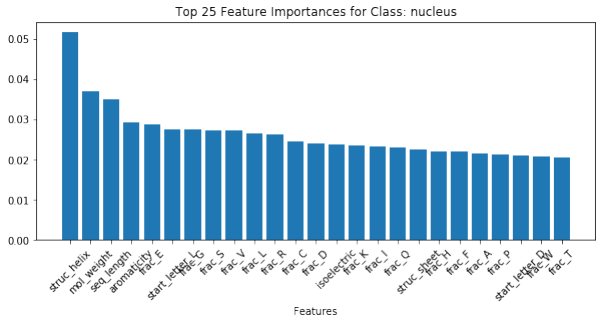
5.4 Further Improvements

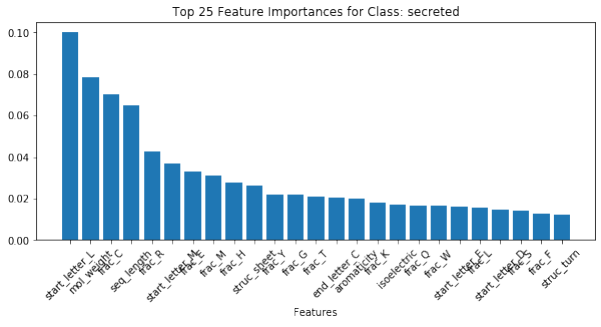
Although this model performed well in distinguishing the secreted and mitochondrial proteins there are a number of additional features that may particularly improve future implementations of this model.

First, some measure of hydrophobicity would be a valuable additional feature. As mentioned in section 2, often the hydrophobic characteristic of signal peptides is what is detected by receptors, but this model only captures these features indirectly, through looking at frequencies of individual amino acids. A specific feature for hydrophobicity wasn't added due to the lack of there being an available hydrophobicity scale in the









BioPython module, but it is likely some measure of this (both over the whole molucule and just the N terminus), would be of some value.

Secondly, it might be worth exploring and searching for specific target patterns within the model. In exploring potential features, I attempted to search for these by creating a feature for all possible "bigrams" and "trigrams" of residues, and examining importances after training, but the high dimensionality made learning difficult and slow. Instead, if one had specific examples of known target peptides this feature would likely greatly help classification, without adding too much to computational complexity. A drawback would be that these examples may be very incomplete, as many signals are still unknown, so there may be a limited benefit.

Finally it may be worth exploring the use of neural networks, as classifiers. Neural networks have shown significant promise in various fields in bioinformatics including protein function prediction **Jones et al**, and could theoretically be applied to this problem. In this case, one could use a richer set of features and feed through a multiclass deep fully connected neural network, or even augment the sequence data with features and the feed augmented sequence into recurrent neural network, such as an LSTM. This latter architecture might have the advantage of allowing investigators to detect new target peptides, through spotting sequences that radically increase likelihoods of a certain classification. However, a draw back is that these neural methods often require a great deal of data, and the dataset here is relatively small so a neural model may not generalise well. This method should be considered, if more training sequence is avaliable.

6 Conclusions

yip

Obtained data from X files, and mixed, shuffle and split into a train and validation split. Since the proteins are all non homologues, no need to arrange by family and could split anyway.

Then took these files and processed them into feature vectors, using the bipython module. These extracted, x features from the sequence.

Then trained a random forest classifier with X trees, and also tested a XGBboost classifier.

Cross validated grid search done to find best parameters for trees Ablation study used to investigate salient features.

Trained a random forest with X trees.

What did you do? - extract features - train -test split - homologies - cross validate - compared with a model (gradient boosting?) - ablation study

- Results of accuracy - Confusion matrix - ablation study - results compared to xgboost

7 Discussion

- Examine the features - Examine the ablation study - Examine comparison to XGBoost - Compare with existing literature and approaches

8 Further Work

- suggestions: more data? better features/

9 Conclusion

10 Theoretical Background

What about ???

end of paper

Table 5. This is table caption

head1	head2	head3	head4
row1	row1	row1	row1
row2	row2	row2	row2
row3	row3	row3	row3
row4	row4	row4	row4

This is a footnote

Fig. 1. Caption, caption.

Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text. Figure 1 shows that the above method Text Text Text Text Text Text Text Text Text Text Text Text. (Bag *et al.*, 2001) wants to know about …… text follows.

$$\sum x + y = Z \tag{1}$$

11 Approach

Equation (1) Text Text Text Text Text Text Text Text TextText. Figure 2 shows that the above method Text Text Text Text Text Text Text Text Text Text Text Text Text. .....

12 Methods

. Figure 2 shows that the above method . text

- for bulleted list, use itemize
- for bulleted list, use itemize
- for bulleted list, use itemize

. Figure 2 shows that the above method .
Text Text Text Text.

1. this is item, use enumerate
2. this is item, use enumerate
3. this is item, use enumerate

Text Text Text Text Text Text Text Text. Figure 2 shows that

the Text Text Text Text Text Text Text Text Text Text Figure 2 shows that the above method Text Text Text Text Text Text Text Text Text Text Text. Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text.

Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text. Figure 2 shows that the above method Text Text Text Text

Acknowledgements

References

Alberts, *et al*. (2008) Molecular Biology of the Cell (5th ed.), *Book Title*, 2nd edn. Publisher, Location, Vol. 1, pp. ???–???.

Bofelli,F., Name2, Name3 (2003) Article title, *Journal Name*, **199**, 133-154.

Bag,M., Name2, Name3 (2001) Article title, *Journal Name*, **99**, 33-54.

Yoo,M.S. *et al*. (2003) Oxidative stress regulated genes in nigral dopaminergic neurnol cell: correlation with the known pathology in Parkinson's disease. *Brain Res. Mol. Brain Res.*, **110**(Suppl. 1), 76–84.

Lehmann,E.L. (1986) Chapter title. *Book Title*. Vol. 1, 2nd edn. Springer-Verlag, New York.

Crenshaw, B.,III, and Jones, W.B.,Jr (2003) The future of clinical cancer management: one tumor, one chip. *Bioinformatics*, doi:10.1093/bioinformatics/btn000.

Auhtor,A.B. *et al*. (2000) Chapter title. In Smith, A.C. (ed.), *Book Title*, 2nd edn. Publisher, Location, Vol. 1, pp. ???–???.

Bardet, G. (1920) Sur un syndrome d'obesite infantile avec polydactylie et retinite pigmentaire (contribution a l'etude des formes cliniques de l'obesite hypophysaire). PhD Thesis, name of institution, Paris, France.