Machine Learning for NLP

Grzegorz Chrupała and Nicolas Stroppa

Saarland University
Google

META Workshop

Outline

Goal of the tutorial

When you leave, we hope you will:

- Be familiar with main ML approaches, principles and ideas
- Know how to apply usual techniques to common problems
- Master dedicated *vocabulary*
- Come up with ideas related to your projects
- Be a bit tired...

Disclaimer

- You might be already familiar with a number of things covered in this tutorial
- We tried to tell a consistent story instead of copying a textbook
- This is a tutorial by dummies and for everyone

Program

Monday, Oct. 18

- 10am-11h30am: Introduction and text classification
- 11h45am-1pm: Classification (part 1)
- 2h30pm-3h30pm: Classification (part 2)
- 3h45pm-5pm: Sequence labeling (part 1)

Tuesday, Oct. 19

- 10am-11h30am: Sequence labeling (part 2)
- 11h45am-1pm: Theoretical and practical matters
- 2h30pm-3h30pm: Open session and exercises
- 3h45pm-5pm: Open session and exercises

Outline

What are you/we doing here? (The Pitch)

Here's the situation.

- You are an employee working in a news aggregation company
- Your main news provider used to assign a category to each news you receive (sports, politics, etc.) but stopped doing it
- Your company still wants this info
- You are the one chosen for this task, that is:
  - *Classify* each incoming news into one of a list of predefined *categories* or *labels*

What are you/we doing here? (The Pitch)

Here's the situation.

- You are an employee working in a news aggregation company
- Your main news provider used to assign a category to each news you receive (sports, politics, etc.) but stopped doing it
- Your company still wants this info
- You are the one chosen for this task, that is:
  - *Classify* each incoming news into one of a list of predefined *categories* or *labels*

Along the way of solving this task, we'll familiarize ourselves with a series of ML techniques

Why you?

Why are you chosen to solve this task?
(aka is that really Natural Language Processing?)

Because:

- You are the only one not associating morphology with body-building
- You know Zipf is not a rock band
- You don't think Katakana is a motorbike brand

Brainstorming session

Assuming you need to solve this task *quickly*, what would you do?
How would you approach the problem?

Brainstorming session

Assuming you need to solve this task *quickly*, what would you do?
How would you approach the problem?

Some technical details about news

- Each news (*document*) is about 300 word long
- They contain a short title
- They are written in English, files are utf8 encoded unicode
- We need to classify 10 news per second
- Targeted market is Europe

Possible ideas

- . . .

Outline

Keywords/Triggers lists

Lists of well-chosen keywords appearing in news or their titles can form very powerful signals.

- Sports: Contador, marathon, Liverpool, scores, Ferrari...
- Politics: Obama, government, War, vote, Ban Ki-moon, reform...
- Entertainment: Lady Gaga, movies, star, Harry Potter, gossip, show...

Keywords/Triggers lists

How can we convert these lists into an actual algorithm?

- If news contains words from list y, then assign label y

Issues with this approach?

Keywords/Triggers lists

How can we convert these lists into an actual algorithm?

- If news contains words from list y, then assign label y

Issues with this approach?

1. Rules can conflict
2. Building accurate lists is difficult

Conflicting rules

Solution for conflicting rules?

Conflicting rules

Solution for conflicting rules?

Idea: we can build different lists with various "priorities":

- Sport-1, Sport-2, Sport-3, Politics-1, Politics-2...

Algo becomes:

- If news contains words from list y-i, then assign label y.
- In case of conflict, assign label with smaller i
- In case of remaining conflict, assign random label among conflicting categories...

Conflicting rules

Solution for conflicting rules?

Idea: we can build different lists with various "priorities":

- Sport-1, Sport-2, Sport-3, Politics-1, Politics-2...

Algo becomes:

- If news contains words from list y-i, then assign label y.
- In case of conflict, assign label with smaller i
- In case of remaining conflict, assign random label among conflicting categories...

- We just moved one level further, we still have to deal with conflicts...
- We also made the process of building lists much more complex...

Where ML comes in

Machine Learning gives sound and theoretically-rooted principles for:

- *Automatically* defining/learning, *from data*, *strategies for solving rule conflicts*

Examples of rules (Part-of-Speech Tagging)

- if token ends with 'ing' and is preceded by token 'is'
  - $\Rightarrow$ label = Verb present participle
- if token = 'both'
  - $\Rightarrow$ label = Adverb
- if token is unknown and starts with a capital
  - $\Rightarrow$ label = Proper Noun
- if previous label = Adjective
  - $\Rightarrow$ label = Noun

Examples of rules (Named-Entity Recognition)

- if token = 'Obama'
  - $\Rightarrow$ label = Person Name
- if token in city list
  - $\Rightarrow$ label = City
- if token matches the regexp '[A-Za-z]+[0-9]+' and previous label = Brand
  - $\Rightarrow$ label = Product model

Examples of rules (Machine Translation)

- the table
  - $\Rightarrow$ la table
- make X up
  - $\Rightarrow$ inventer X
- either NP1 or NP2
  - $\Rightarrow$ soit NP1 soit NP2

Examples of rules (Machine Translation)

- the table
  - $\Rightarrow$ la table (phrase-based)
- make X up
  - $\Rightarrow$ inventer X (hierarchical phrase-based)
- either NP1 or NP2
  - $\Rightarrow$ soit NP1 soit NP2 (syntax-based)

Outline

Where ML comes in

Machine Learning gives sound and theoretically-rooted principles for:

- *Automatically* defining/learning, *from data*, *strategies for solving rule conflicts*

Data Representation - Text classification

What ML is not doing is defining what those rules represent, i.e. *you* need to decide on a *data representation*.

For text classification, we implicitly assumed a *bag-of-words* representation, i.e. a vector **x** whose *dimensions* are words/unigrams, and the associated values are the number of occurences of those words in document x.

The data representation must be:

- *expressive* enough to allow you define interesting rules for your problem
- *simple* enough to allow models to be learned

Data Representation - Text classification

What ML is not doing is defining what those rules represent, i.e. *you* need to decide on a *data representation*.

For text classification, we implicitly assumed a *bag-of-words* representation, i.e. a vector $\mathbf{x}$ whose *dimensions* are words/unigrams, and the associated values are the number of occurences of those words in document x.

The data representation must be:

- *expressive* enough to allow you define interesting rules for your problem
- *simple* enough to allow models to be learned, in a tractable and efficient way

$\Rightarrow$ Obviously, sometimes, tradeoffs must be made...

Data Representation - Sequence Labeling

In *Sequence Labeling*, we usually represent the input and the output as sequences (size = sentence length).

We usually want rules to be applicable to tokens and their neighbours.

Data Representation - Machine Translation

In Machine Translation, we usually represent the input and the output as sequences (size = sentence length).

The representation must also take into account *hidden* structures such as:

- alignments
- parse trees
- reordering models

Again, there is often a tradeoff to make between expressiveness and simplicity/efficiency...

Know your Data!

- Chosing a proper data representation is crucial
- This choice is application dependent
- Can only be done by an expert that knows which rules are useful and need to be defined...
- This is a vision of the world, different visions/views can be combined
- Be aware of the implications of the approximations you're making

Know your Data!

- Chosing a proper data representation is crucial
- This choice is application dependent
- Can only be done by an expert that knows which rules are useful and need to be defined...
- This is a vision of the world, different visions/views can be combined
- Be aware of the implications of the approximations you're making
- The learning algo itself (almost) doesn't matter...(don't say it loud)

Know your Data!

- Chosing a proper data representation is crucial
- This choice is application dependent
- Can only be done by an expert that knows which rules are useful and need to be defined...
- This is a vision of the world, different visions/views can be combined
- Be aware of the implications of the approximations you're making
- The learning algo itself (almost) doesn't matter... (don't say it loud)

Personal note: I've seen a number of ML projects failed simply because some people thought ML could automagically understand the data in their place...

Rule-based approaches and ML

Machine Learning gives sound and theoretically-rooted principles for:

- *Automatically* defining/learning, *from data*, *strategies for solving rule conflicts*

Hmm. . . Wait. . . What about Rule-based approaches vs. Machine Learning?

Rule-based approaches and ML

Machine Learning gives sound and theoretically-rooted principles for:

- *Automatically* defining/learning, *from data*, *strategies for solving rule conflicts*

Hmm. . . Wait. . . What about Rule-based approaches vs. Machine Learning?

- Ad-hoc vs. Sound rule conflict resolution
  - "Hard" vs. "Soft" rules
- Manually vs. Meta-manually constructed rules (automatically generated rules)
- Usually different expressiveness/efficiency tradeoff

Keywords/Triggers lists issues (cont.)

How can we convert these lists into an actual algorithm?

- If news contains words from list y, then assign label y

Issues with this approach?

Keywords/Triggers lists issues (cont.)

How can we convert these lists into an actual algorithm?

- If news contains words from list y, then assign label y

Issues with this approach?

1. Rules can conflict
2. Building accurate lists is difficult

Automatically building lists

Solution for building accurate lists?

- we want to automatically generate those lists
- we manually fix a framework, but the rules are automatically extracted (meta-manually constructed)

Automatically building lists

Solution for building accurate lists?

- we want to automatically generate those lists
- we manually fix a framework, but the rules are automatically extracted (meta-manually constructed)

Idea: leverage *labeled examples*, i.e. news that were previously categorized

We can compute things like the most representative (salient) words/phrases in each category.

Automatically building lists

We can compute things like the most representative (salient) words/phrases in each category.
For example, we can compute conditional entropy:

$$H(Y|X) = -\sum_{l} p(l|X) \log p(l|X),$$

where l is the output label and X a variable corresponding to "document contains a given word w".
(Related to Information gain through $IG(Y|X) = H(Y) - H(Y|X)$)

Data/Rule preparation

By doing a pre-selection or a pre-scoring of the rules, we are performing:

- *Feature selection*
- Feature weighting

By operating a feature selection, we're *reducing the dimensionality* of the input space.
This decreases expressiveness, but increases learnability. So, if performed carefully, will improve quality of the learned models.

Again, this is about knowing your data...

Data/Rule preparation

Such "rule preparations" are really important.
In the case of news classification, we can also mention things like

- tokenization (phrase? multi-word exp? CJK languages? ...)
- spelling normalization
- orthographic normalization
- morphological normalization and variants
- specific rules for titles
- entity recognition
- TF-IDF-like transformations (cf. Zipf law)
- ...

This can be considered pre-processing, but it also can be considered part of the data representation considerations.

Solving conflicts with linear models

Machine Learning gives sound and theoretically-rooted principles for:

- *Automatically* defining/learning, *from data*, *strategies for solving rule conflicts*

For example, in *linear models*, a weight/score is automatically assigned to each rule, scored are summed over all rules (hence linear), and conflicts are solved by taking the final higher score.

Solving conflicts with linear models

Machine Learning gives sound and theoretically-rooted principles for:

- *Automatically* defining/learning, *from data*, *strategies for solving rule conflicts*

For example, in *linear models*, a weight/score is automatically assigned to each rule, scored are summed over all rules (hence linear), and conflicts are solved by taking the final higher score.

But Grzegorz will explain it better...