Automatic Processing of User-Generated Short Texts and applications to Conversational Systems

Lara, the Meetic Chatbot

Thomas Dopierre
May 10, 2021

Laboratoire Hubert Curien
Meetic
Introduction - Meetic & Lara
Introduction – Meetic

- Dating website created in 2001
- At the origin of more than 8 millions of couples
- More than 5 billion interactions between users in 2014
- Present in 15 countries, 13 languages
- First dating website to have a Chatbot

1https://www.meetic-group.com/who-we-are/
Born in 2017, *Lara* is a conversational agent, with multiple goals.
Born in 2017, *Lara* is a conversational agent, with multiple goals.

Profile Creation

**Hi, I’m your Match coach. How can I help you?**

- I am looking for a man
- I am looking for a woman
Born in 2017, *Lara* is a conversational agent, with multiple goals:

**Profile Creation**

**Customer Service**
Lara – Goals

Born in 2017, Lara is a conversational agent, with multiple goals

Profile Creation

Customer Service

Profile Recommendation
Every day, Lara

- Talks to more than **50,000** users, in 6 languages (English, French, German, Spanish, Italian, Dutch)
- Presents more than **70,000** profiles
I lost my password

No problem! We'll try to recover it.

Figure 1: Simplified overview of the Lara Framework
Lara – Challenges

- Understand more than 300 different user intents
- User intents usually evolve with time
- Learn to classify using **very few samples**

**Good news:** As users interact with the bot, we have some unlabeled data to work with!
Sentence Encoders
Sentence Encoding

From raw text...

I have lost my password

I don't like his hair

Do you have a profile for me?

How to recover my password?

...To an embedding space
Sentence encoders have evolved in the last years

- Fixed word embeddings \{GloVe [17], FastText [10], Dict2Vec [25]\} + model \{CNN [11], Bi-LSTM [14], GRU [30]\} to embed sentences
- Contextualised embeddings with ELMo [18]
- Transformers [26]
Sentence encoders have evolved in the last years

- Fixed word embeddings\{GloVe [17], FastText [10], Dict2Vec [25]\} + model \{CNN [11], Bi-LSTM [14], GRU [30]\} to embed sentences
- Contextualised embeddings with ELMo [18]
- Transformers [26]

Transformers have been a major break point for many tasks [3]
Choosing the right sentence encoder

Fine-tuning sentence embeddings even on small task-specific dataset can lead to large performance gains [23, 9]
Fine-tuning sentence embeddings even on small task-specific dataset can lead to large performance gains [23, 9]

We want to use BERT, so we fine-tune it on the masked language modeling task for a few epochs on the different datasets.
Choosing the right sentence encoder

Fine-tuning sentence embeddings even on small task-specific dataset can lead to large performance gains [23, 9]

We want to use BERT, so we fine-tune it on the masked language modeling task for a few epochs on the different datasets.

MLM is fully unsupervised, so we can use the whole datasets!
Choosing the right sentence encoder

Figure 2: T-SNE representations of the Snips [2] and OOS [12] datasets using different encoders
Few-Shot Learning vs. Pseudo-Labeling
A problem overview

"I lost my password"

We have both labeled and unlabeled data at hand

| LABELED | ≪ | UNLABELED |

We want to train a classifier using labeled (and maybe unlabeled) data

Figure 3: Data overview
### Pseudo-Labeling

**Goal:** Assign pseudo-labels to unlabeled data, train a model using both labels and pseudo-labels

**Evaluation Metric:** correctness of pseudo-labels & Intent Detection performance

---

**Figure 4:** Pseudo-Labeling Framework

<table>
<thead>
<tr>
<th>Labeled</th>
<th>Unlabeled</th>
</tr>
</thead>
<tbody>
<tr>
<td>password_lost</td>
<td>I lost my password, can you help?</td>
</tr>
<tr>
<td>see_profiles</td>
<td>Do you have any more profiles? Hey can you find someone for me?</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>??</td>
<td>I want to stop receiving e-mails from you I can't remember my password ...</td>
</tr>
</tbody>
</table>

---

**Sentence Embedder**

**Generate pseudo-labels for unlabeled data**

---

**Intent detection**

<table>
<thead>
<tr>
<th>Labeled</th>
<th>Pseudo-labeled</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Labeled" /></td>
<td><img src="image" alt="Pseudo-labeled" /></td>
</tr>
</tbody>
</table>
Few-Shot Learning as a Meta-Learning process

Figure 5: Episode-like process in Few-Shot Image Classification

- Episode-like process [22, 28], where support points are used to predict query points
- Train, valid and test labels are different

Goal: Train a end-to-end model, which will be able to generalise to test classes using only few shots
**Idea:** Represent each class as the average vector (prototype) of its support points.

**Prediction:** The label assigned to a query point $x$ is the one for which the euclidean distance to the prototypes is the lowest.

*Figure 6: Prototypical Networks*
Other Few-Shot Learning Methods

Matching Network [28]

Prototypical Network [22]

Relation Network [24]

Induction Network [8]

Baseline Network

Baseline++ Network [1]

Figure 7: Few-Shot Text Classification methods used in our experiments
We have contributions on both approaches
In this presentation

We have contributions on both approaches

• A novel method for pseudo-labeling using hierarchical clustering
In this presentation

We have contributions on both approaches

- A novel method for pseudo-labeling using hierarchical clustering
- An updated benchmark of end-to-end few-shot text classification methods using a common text encoder
We have contributions on both approaches

- A novel method for pseudo-labeling using hierarchical clustering
- An updated benchmark of end-to-end few-shot text classification methods using a common text encoder
- A novel extension of Prototypical Networks using unlabeled data and diverse paraphrasing
Pseudo-Labeling with Hierarchical Clustering
Idea: Use hierarchical clustering to produce pseudo-labels for unlabeled data points

Goals: Avoid using too much hyper-parameters. Avoid assigning pseudo-labels to all data points, (some unlabeled data points might be noise)

Figure 8: Examples of clusters obtained by hierarchical clustering

\(^1\)https://scikit-learn.org/stable/modules/clustering.html
Pseudo-Labeling with Hierarchical Clustering – Step (a)

**Input:** Labeled and unlabeled data points

**Process:** Iteratively constructs a tree using hierarchical clustering. Binds the two closest clusters (according to Ward’s method) at each iteration.

**Output:** A tree structure, where each non-leaf node has exactly two children, representing a merge between two nodes at some iteration.

*Figure 9: Two-fold Pseudo-Labeling: Step (a)*
**Process:** Iteratively de-constructs the tree obtained at step (a), splitting clusters into their two sub-clusters until (1) all generated clusters contain either no labeled data points, or (2) some labeled data points with a unique label.

**Case (1)** Discard data points.

**Case (2)** Assign the unique label to all unlabeled data points in the same cluster.

Figure 10: Two-fold Pseudo-Labeling: Step (b)
### Table 1: Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#sentences</th>
<th>#classes (train/valid/test)</th>
<th>#sentences/class</th>
<th>#tokens/sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Snips [2]</td>
<td>14,484</td>
<td>2/2/3</td>
<td>2,069 ± 21</td>
<td>9.0 ± 3.2</td>
</tr>
<tr>
<td>OOS [12]</td>
<td>23,700</td>
<td>53/52/46</td>
<td>157 ± 85</td>
<td>8.5 ± 3.3</td>
</tr>
<tr>
<td>Liu [15]</td>
<td>25,478</td>
<td>19/18/17</td>
<td>472 ± 823</td>
<td>7.5 ± 3.4</td>
</tr>
<tr>
<td>R8 [19]</td>
<td>7,685</td>
<td>3/2/3</td>
<td>961 ± 1303</td>
<td>102 ± 117</td>
</tr>
</tbody>
</table>
Table 2: F1-score of Intent Detection system fed with pseudo-labels from various methods.

According to experiments, our method:

- is the best on all datasets
- is robust to the large number of classes (OOS) and the more general case of text classification (R8)

This work was published as long paper at COLING2020: Few-shot Pseudo-Labeling for Intent Detection [6]
End-To-End Few-Shot Learning Methods
Motivation: Find an End-To-End Few-Shot Learning method which takes into account unlabeled data

First: We need to study state-of-the-art Few-Shot Text Classification methods!

Problem: Methods in the literature do not use the same sentence encoder...

<table>
<thead>
<tr>
<th>Model</th>
<th>Mean Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matching Networks [28]</td>
<td>65.73</td>
</tr>
<tr>
<td>Prototypical Networks [22]</td>
<td>68.17</td>
</tr>
<tr>
<td>Graph Network [20]</td>
<td>82.61</td>
</tr>
<tr>
<td>Relation Network [24]</td>
<td>83.07</td>
</tr>
<tr>
<td>SNAIL [16]</td>
<td>82.57</td>
</tr>
<tr>
<td>ROBUSTTC-FSL [29]</td>
<td>83.12</td>
</tr>
<tr>
<td>Induction Networks [8]</td>
<td>85.63</td>
</tr>
</tbody>
</table>

Table 3: Reported Results of Few-Shot Text Classification models on the ARSC dataset

Idea: Compare methods using the same transformer-based encoder
### Few-Shot Learning: A reality check on ARSC

<table>
<thead>
<tr>
<th>Model</th>
<th>Configuration</th>
<th>Mean binary accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Metric</td>
<td>Original encoder †</td>
</tr>
<tr>
<td>Matching Network</td>
<td>euclid.</td>
<td>–</td>
</tr>
<tr>
<td>(Vinyals et al., 2016)</td>
<td>cosine</td>
<td>65.7</td>
</tr>
<tr>
<td>Prototypical Network</td>
<td>euclid.</td>
<td>68.2</td>
</tr>
<tr>
<td>(Snell et al., 2017)</td>
<td>cosine</td>
<td>–</td>
</tr>
<tr>
<td>Proto++</td>
<td>euclid.</td>
<td>–</td>
</tr>
<tr>
<td>(Ren et al., 2018)</td>
<td>cosine</td>
<td>–</td>
</tr>
<tr>
<td>Relation Network</td>
<td>N/A</td>
<td>–</td>
</tr>
<tr>
<td>(Sung et al., 2018)</td>
<td>base</td>
<td>–</td>
</tr>
<tr>
<td>Induction Network</td>
<td>N/A</td>
<td>ntl</td>
</tr>
<tr>
<td>(Geng et al., 2019)</td>
<td>N/A</td>
<td>–</td>
</tr>
<tr>
<td>Baseline</td>
<td>N/A</td>
<td>–</td>
</tr>
<tr>
<td>Baseline++</td>
<td>euclid.</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>cosine</td>
<td>–</td>
</tr>
</tbody>
</table>

**Figure 11:** Reality-check of Few-Shot Learning systems on the ARSC dataset
Table 4: Few-Shot Learning methods, tested on 3 intent detection datasets, provided the same transformer encoder.

Main findings:

- Prototypical Networks reclaim the state-of-the-art when equipped with a transformer
- The distance metric plays an important role
- A traditional classifier yields competitive results

This work was published as long paper EACL2021: A Neural Few-Shot Text Classification Reality Check [4]
PROTAUGMENT: Intent Detection
Meta-Learning through
Unsupervised Diverse
Paraphrasing
Motivations

- Extend prototypical networks