Exploring Attention for Keyword Spotting

A review

Riccardo Mazzieri

***Abstract*—Abstract goes here:**

***Index Terms*—Self Organizing Maps, Unsupervised Learning, Optimization, Neural Networks, Recurrent Neural Networks. A list of keywords defining the tools and the scenario. I would not go beyond *six* keywords.**

I. Introduction

The Keyword Spotting (KWS) task consists in the developement of systems which are able to recognize the presence of a certain predetermined set of keywords from a stream of user utterances. In recent years, this problem has become increasingly popular and, with the rapid development of mobile devices, it is now playing a vital role in human-computer interaction and encouraging the adoption of hands-free interfaces for a wide variety of use cases, that can range from "smart home" devices like Amazon Alexa, to virtual assistants such as Apple's Siri or Google Assistant. KWS systems play a complementary role with respect to more complex, cloud based speech recognition services: indeed, their are required to run on local devices and are often used to initiate interactions by pronouncing the correct keyword. For this reason the research in KWS typically focuses on a tradeoff between highly precise predictions and small memory/computational footprint.

At present, Deep Learning techinques represent the state of the art approach to the KWS problem and have shown to give good results on the performance/footprint tradeoff [1], [2]. Research on this field is active and novel architectures are continuously being proposed. The most common class of types that tackle the KWS problem are Convolutional Neural Networks (CNN), and Recurrent Neural Netoworks (RNN).

II. Related Work

Some hints:

- **Goal:** The goal of this section is to describe what has been done so far in *the* literature. You should focus on and briefly describe the work done in the best papers that you have read.
- **Length:** One full column is fine but often this takes one column and a half. It is very easy to use a full page, although this may just due to your sloppiness... if you carefully go through the one page long version, you often find it possible to compact it in one column and a half. In any event, I would make this section no longer than one page, this leads to an overall *two pages* including abstract, introduction and related work. I believe this is a fair amount of space in most cases.
- **Approach:** For each you should comment on the paper's contribution, on the good and important findings of such paper and also, **1)** on why these findings are not enough and **2)** how these findings are improved upon/extended by the work that you present here. At the end of the section, you may recap the main paper contributions (maybe one or two, the most important ones) and how these extend/improve upon previous work.
- **References:** please follow this *religiously*. It will help you a lot. Use the Latex `Bibtex` tool to manage the bibliography. A `Bibtex` example file, named `biblio.bib` is provided with this template.

- **Citing conference/workshop papers:** I recommend to always include the following information into the corresponding `bibitem` entry:
  1) author names,
  2) paper title,
  3) conference / workshop name,
  4) conference / workshop address,
  5) month,
  6) year.

  Examples of this are: [**?**] [**?**].

- **Citing journal papers:** I recommend to always include the following information into the corresponding `bibitem` entry:
  1) author names,
  2) paper title,
  3) full journal name,
  4) volume (if available),
  5) number (if available),
  6) month,
  7) pages (if available),
  8) year.

  Examples of this are: [**?**] [**?**] [**?**].

- **Citing books:** I recommend to always include the following information into the corresponding `bibitem` entry:
  1) author names,
  2) book title,
  3) editor,
  4) edition,
  5) year.

Remark II.1. Note that some of the above fields may not be shown when you compile the Latex file, but this depends on the bibliography settings (dictated by the specific Latex style that you load at the beginning of the document). You

may decide to include additional pieces of information in a given bibliographic entry, but please, **be consistent** across all the entries, i.e., use the same fields for the same publication type. Note that some of the fields may not be available (e.g., the paper *volume*, *number* or the *pages*).

III. Processing Pipeline

Remark III.1. On tailoring the paper structure to your needs: The structure recommended for the previous sections is rather standard and could work for different papers with differing technical content, the structure and the paper content from here on highly depends on the type of paper, possibilities are: mostly based on theoretical analysis, showing experimental design/activity, proposing a new technique and analyzing its performance via experiments or simulation. For the HDA course we deal with machine learning and, in detail, with training and testing neural network architectures to perform some specific inference or classification task. The following structure and comments are specifically addressing this type of technical content.

Remark III.2. Why having this section: With this section, we start the technical description with a *high level* introduction of your work (e.g., processing pipeline). Here, you do not have to necessarily go into the technical details of every block/algorithm of your design, this will be done later as the paper develops. What I would like to see here is a high level description of the approach, i.e., which processing blocks you used, what they do (in words) and how these were combined, etc. This section should introduce the reader to your design, explain the different parts/blocks, how they interact and why. You should not delve into technical details for each block, but you should rather explain the big picture. Besides a well written explanation, I often use a nice diagram containing the various blocks and detailing their interrelations.

Writing tips: Sections, Figures and Tables are usually shortened as Sec., Fig., Tab.

- **Cross referencing:** In Latex, cross referencing is easy. You need to label an object through the `\label{labelid}` command and referencing it where you need it through the `\ref{labelid}` command.
- **Suggestion:** I suggest to cross reference a table using `Tab.~\ref{tab:tableid}`, the same holds for figures and sections, by just replacing "`Tab.`" with "`Fig.`" and "`Sec.`". Of course when defining the table, you need to add a Latex command `\label{tab:tableid}` in the right place inside the table Latex environment to cross-link it. The tag `tab:tableid` is user defined and is the identifier that you associate with the table in question. You could call it `pippo` of you wish, but I recommend to use something like "`tab:tableid`" for tables, "`eq:eqid`" for equations, "`sec:secid`" for sections and so forth, where `tableid` has to be unique for each table in the document. The same applies to figures and all other objects. I guess you got the idea. This will lead to a neat Latex code and will facilitate cross referencing while avoiding duplicate labels, especially in large Latex documents (think of a book for example).
- **But what about the tilde?** This is a nice trick I have learned from a friend (many years ago from Prof. Frank Fitzek, now at TU Dresden). When you write `Tab.~\ref{tab:tableid}` Latex knows that `Tab.` and the corresponding table number `\ref{tab:tableid}` must be displayed within the same line, i.e., they can never be broken across lines. This is nice and desirable I believe. I always use it for all referenced material, including citations; example: "As done in~`\cite{suppa-wu-2019}`."
- **More about breaking stuff across lines** often times you have composed words, in line equations, etc. and for some reason you would like Latex to never break them across lines. Example: the Latex command `\mbox{Neural Networks (NN)}` is processed by Latex so that "Neural Networks (NN)" is never broken across lines. I use this very often, also for inline equations that I do not want Latex to split across subsequent lines.

IV. Signals and Features

Being a machine learning paper, I would have here a section describing the signals you have been working on. If possible, you should describe, in order,

1) the measurement setup (if relevant), how input data is formatted (e.g., vectors of fixed size measured at regular sampling times),
2) how the signals were pre-processed, e.g., to remove noise, artifacts, fill gaps (missing points) or to re-interpolate the signals to represent them through a constant sampling rate, etc.
3) after this, you should describe how *feature vectors* were obtained from the pre-processed signals. If signals are *time series* this also implies stating the segmentation / windowing strategy that you adopted, to then describe how you obtained a feature vector for each time window. Also, if you use existing feature extraction approaches, you may want to briefly describe them as well, in addition to (and before) your own (possibly new) feature extraction method.

Last but not least, this section should also contain information on how you have split the dataset into training, validation and test sets. This will be briefly recalled within the "Results" section.

V. Learning Framework

Here you finally describe the learning strategy / algorithm that you conceived and used to solve the problem at stake. A good diagram to exemplify how learning is carried out is often very useful. In this section, you should describe the learning model, its parameters, any optimization over a given parameter set, etc. You can organize this section into sub-sections. You are free to choose the most appropriate structure.

Remark V.1. Note that the diagram that you put here differs from that of Section III as here you show the details of how your learning framework, or the core of it, is built. In Section III you instead provide a high-level description of the involved processing blocks, i.e., you describe the *processing flow* and the rationale behind it.

On math typesetting: there are many Latex tricks that you should use to produce a high quality technical essay. A few are listed below, in random order:

- **Vectors and matrices:** x is a scalar, whereas $\boldsymbol{x}$ (in bold) is a vector, and $\boldsymbol{X}$ is a matrix with elements $\boldsymbol{X} = [x_{ij}]$. For bold symbols you may use the `\bm` Latex command, e.g., `\bm(x)`.
- **Operators:** such as max, min, argmax, argmin and special functions such as $\log(\cdot)$, $\exp(\cdot)$, $\sin(\cdot)$, $\cos(\cdot)$ are obtained through specific latex commands `\min`, `\max`, `\arg\!min`, `\arg\!max`, `\log`, `\exp`, `\sin`, `\cos`, etc. Use them! log, exp, min, sin, cos, etc., look ugly.
- **Sets** can be represented through calligraphic fonts, e.g., $\mathcal{S}$, $\mathcal{F}$, $\mathcal{B}$, etc., obtained using the Latex command `\mathcal{S}`, etc.
- **Equations:** for a single equation use the `equation` Latex environment. Example:

$$\sigma(x) = \frac{1}{1+e^{-x}}. \tag{1}$$

Now, using round brackets `(` and `)` we get

$$\sigma(x) = (\frac{1}{1+e^{-x}}), \tag{2}$$

but this looks ugly, you should use "`\left (`" for "(" and "`\right )`" for ")", obtaining

$$\sigma(x) = \left(\frac{1}{1+e^{-x}}\right). \tag{3}$$

- **Punctuation:** Displayed equations are usually considered to be part of the text and, in turn, they will get the very same punctuation as if they were inline with the text (and part of the sentence). If the sentence ends with a displayed equation, the equation gets a period "." right after it, see Eq. (1). iI the equation is instead part of a running sentence, which is continued after it, then the equation may be ended by a "," as in Eq. (2). Use the standard grammar rules and your good sense of flow to assess how equations should be punctuated, I usually read through as if they were plain text.

VI. Results

In this section, you should provide the numerical results. You are free to decide the structure of this section. As a general "rule of thumb", use plots to describe your results, showing, e.g., precision, recall and F-measure as a function of the system (learning) parameters. You can also show the precision matrix.

Remark VI.1. Present the material in a progressive and logical manner, starting with simple things and adding details and explaining more complex findings as you go. Also, do not try to explain/show multiple concepts within the same sentence. Try to **address one concept at a time**, explain it properly, and only then move on to the next one.

Remark VI.2. The best results are obtained by generating the graphs using a vector type file, commonly, either `encapsulated postscript (eps)` or `pdf` formats. To plot your figures, use the Latex `\includegraphics` command. Lately, I tend to use pdf more.

Remark VI.3. If your model has hyper-parameters, show selected results for several values of these. Usually, tables are a good approach to concisely visualize the performance as hyper-parameters change. It is also good to show the results for different flavors of the learning architecture, i.e., how architectural choices affect the overall performance. An example is the use of CNN only or CNN+RNN, or using inception for CNNs, dropout for better generalization or attention models. So you may obtain different models that solve the same problem, e.g., CNN, CNN+RNN, CNN+inception, etc.

VII. Concluding Remarks

This section should take max half a page, I personally find it difficult to come up with really useful observations, I mean ones that bring a new contribution with respect to what you have already expounded in the "Results" section. In case you have some serious stuff to write, you may also extend the section to 3/4 of a page :-).

In many papers, here you find a summary of what done. It is basically an abstract where instead of using the present tense you use the past participle, as you refer to something that you have already developed in the previous sections. While I did it myself in the past, I now find it rather useless.

What I would like to see here is:

1) a very short summary of what done,
2) some (possibly) intelligent observations on the relevance and *applicability* of your algorithms / findings,
3) what is still missing, and can be added in the future to extend your work.

The idea is that this section should be *useful* and not just a repetition of the abstract (just re-phrased and written using a different tense...).

Moreover: being a project report, I would also like to see a specific paragraph stating

4) what you have learned, and
5) any difficulties you may have encountered.

REFERENCES

[1] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4087–4091, 2014.

[2] T. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," in *INTERSPEECH*, 2015.