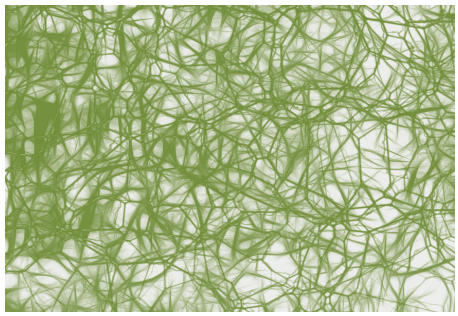
COMSM0018: Applied Deep Learning (2019-20)

Your Team Project

(task for *up to* 3 student teams, summative, 60%)

Re-produce a Published Research Paper



Forming your Teams:
Register your team of up to 3 people (i.e. one, two or three students) online at:
https://doodle.com/poll/5dhfc28m9n9qm4sy
Post registration, teams can split but cannot merge, to avoid any copying of code or ideas

Each member of the team should submit an exact copy of the final submission on Blackboard by the deadline. The report (see below) should note the full names and emails of all members of the team.

It is up to each team to decide their best strategy to tackle this problem, i.e. whether to divide the tasks below, or to work together on all tasks. Contributions of team members need not be explicitly stated.

Task Brief:
This assignment gives you the opportunity to appreciate the work required in replicating published research from a publicly available dataset and manuscript. It allows you to reflect on the experience of reproducing published results and potentially outperforming on your replication.

Gathering all the knowledge you acquired from the lectures and labs, read the paper below carefully and replicate the required results (Note: you are not required to re-produce all the paper's results). Feel free to take any pieces of code from the labs as a baseline, but the rest of the code should be originally yours.

The Paper:

Y Su, K Zhang, J Wang and K Madani. Environment Sound Classification using a Two-Stream CNN Based on Decision-Level Fusion. In Sensors 19 (7), 2019. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6479959/pdf/sensors-19-01733.pdf

Note that our choice for paper is based on its simplicity and similarity to your labs, rather than its superior performance or exceptional novelty.

Please read the following information carefully before attempting the replication.

1) Train/Test Folds:

Do **NOT** use the 10-fold approach in the paper – this is too long to train. Instead we offer you a specified single train/test fold.

2) Input Features:

Several 2D features are extracted from the raw waveforms in the dataset, which represent frequency-time, namely: log-mel spectrograms, Mel-Frequency Cepstral Coefficients (MFCC), chroma, spectral contrast, and tonnetz. These 2D features can be seen as images and be fed at convolutional networks for training. We are providing you with these features pre-calculated and ready to use for training.

The main contributions of this paper are: 1) a convolutional network architecture for training with such features, and 2) the fusion of these features in different ways.

3) CNN Architecture:

Three variations of the architecture are noted in the paper, 4-conv (main architecture), 6-conv and 8-conv layers architectures for comparison.
You have to implement **ONLY** the 4-conv layer architecture.

The architecture is a shallow convolutional network with a few convolutional and fully-connected layers. **HOWEVER**, we spotted some inconsistencies in the description of the architecture in the paper.

One of your tasks will be to **identify these inconsistencies** as part of your reproduction of this paper (note Section F in your report). We will give you a few hints here:

1. The confusion is in Section 3.2.
2. Figure 2 is correct but not sufficient to fully reimplement the architecture.
3. Table 1 is also correct and contains most of the information needed to build the architecture. Make sure your CNN has the same number of parameters are those noted in this table.
4. Figure 4, in combination with Table 1, can help you understand what is the correct architecture and where the inconsistencies are. **Extra hint**: Although most architecture figures report output shapes for each layer, in Figure 4 the authors are reporting input shapes for each layer.
5. **Final Hint:** there are two correct answers, not a single one. These produce similar results. You are only required to note one.

4) Fusion:

First of all, the authors propose 2 inputs for training which are the result of combining a subset of the individual features mentioned above. The first input is called LMC and the second is MC, which can be seen in Figure 1 in the paper. You are expected to compose these inputs by combining the features given (ref Section 3.1).

Based on those and the 4-conv layer architecture, you should train 2 networks called LMCNet and MCNet, named after the inputs.

In Section 3.3, the authors describe a late-fusion method. This is complex; and **we do NOT expect you to implement this.**

Instead, implement a standard yet effective late fusion approach: LMCNet and MCNet will be trained independently, and then you will combine their predictions during testing. For each example in the test set, calculate the predictions (i.e. the logits) of LMCNet and MCNet, apply softmax to each then average class-wise. This is what you will call **TSCNN. Advice**: For fusing the 2 models, instead of testing both models online, train each model and save the scores then use these later for fusion.

Finally, the authors concatenate all the 5 features (the ones used to construct LMC and MC) in a single input which they call MLMC and they train the 4-layer network using this feature. They do so to compare the performance of late fusion with input-level fusion. You are expected to code and train this additional 4-layer network using the MLMC feature as well. **Hint:** You will need to adjust the size of your first fully-connected layer for this. **Why?**

5) Evaluation:

You will report the performance of 4 different models: LMCNet, MCNet, MLMC, and TSCNN, replicating the results of Table 2, using the single train/test fold given to you.

The dataset was constructed by splitting the audio of each example, into multiple segments. During training, you will treat each segment as an independent training sample. **However,** during testing, you have to combine the predictions of the segments that belong to the same audio file. With each segment, we are providing you with a unique identifier of the audio file that the segment belongs to. You need to compute the score of each audio segment, then average the scores of all the segments that belong to a given audio file, in order to compute the final score of that audio file. This can then be compared to the correct label of the audio file.

6) Our replicated results:
Replicating papers rarely produces exact results as those reported in the published papers. It is highly advisable to publish one's code with the paper, however this is very infrequently adopted by researchers.

We have replicated the paper's results for you in our own version of Table 2, because a) we are using a single fold rather than 10-fold cross-validation, and b) we found inconsistencies in the results reported by the authors. We provide the corresponding table on page 3 (end of this document) that we could re-produce, using the split we provide. These are the results you are attempting to reproduce.

Dataset and Helpful Code:
You can find resources we've prepared for you for this project at: **https://tinyurl.com/y5nnvzmb**

Final Submission:

1. *An original code, **based on PyTorch** (other software engines won't be accepted – we won't accept Keras or Tensorflow), replicating the published paper. You can use your lab code from any or all group members. We aim to run your code on BC4, so ensure it compiles and runs.*

2. *A report in the IEEE conference format (https://www.ieee.org/conferences_events/conferences/publishing/templates.html) of up to 5 pages including references, submitted in PDF format. The report should include the following sections:*

   A. ***Title** and **Team members** (names and usernames)*

   B. ***Introduction**: Definition of the problem addressed by the paper Su et al (in your own words)*

   C. ***Related Work**: A summary of published papers attempting to address the same problem (up to 3 works). These could be from the references of the paper itself, or otherwise.*

   D. ***Dataset**: A description of the dataset used, training/test split size, labels and file formats.*

   E. ***Input**: Explain the LMC and MC inputs used. Give 1-2 examples visually from your data, by plotting these as images. Do not use the figure from the original paper.*

   F. ***Architecture (Su et al)**: Summarise the 4-conv architecture's details, in writing, through a table or a diagram (only one of these). Detail your findings of where the paper's inconsistencies are, and how you went around resolving these.*

   G. ***Implementation Details**: Summary of the steps you have undertaken to replicate the results, train the data and obtain the results, including any decisions you needed to make along the way. Do not include any pieces of code, but you can include pseudo-codes if needed.*

   H. ***Replicating Quantitative Results**: You need to present your results for table 2.*

   I. ***Training curves**: Include your training/test loss (and avg accuracy) curves for your models, and comment on any overfitting in your training. The tables here should correspond to the same run as those in the reported table (Section H). These curves could be directly retrieved from Tensorboard.*

   J. ***Qualitative Results:** This section should include sample success and failure cases based on your algorithm. In presenting these examples, you can plot/display the inputs(s) in each case. Particularly: (a) find one or more examples that are correctly classified by both LMCNet and MCNet. (b) find at least one case where one input is correct while the other is incorrect. (c) find one case where late fusion outperforms individual inputs, (d) find one example where all methods fail.*

   K. ***[65+] Improvements**: Using the same 4-conv layer architecture, propose, implement and test one potential improvement you made to your results (i.e. do not use the 6-conv or 8-conv). **Note:** if you describe multiple improvements, we will give you the lower mark (rather than the higher one), so choose the one you believe in. Cover any implementation details required to understand and replicate your modifications. Report your improved results in tabular format for all metrics. Do not include any pieces of code, but you can include pseudo-codes if needed. **Note:** any improvements should be made using the same dataset, train/test split and evaluation metrics used earlier. Improvements can include changes to architecture, hyper-parameters and learning algorithm. Your choice should be justified theoretically and experimentally.*

   L. ***Conclusion and Future Work**: Summarise what your report contains in terms of content and achievements.*

Suggest future work that might extend, generalise or improve the results in your report.

Deadline:
Deadline for submission via Blackboard is
Wed 15 Jan 2019 17:59
Each member of the team should submit a copy of the final submission (code + report).

Marking Guideline.

Note: *Code and report will be checked for plagiarism. Proven plagiarism will result in a 0 grade on this coursework for the whole team.*

50-54
To pass this assignment, you must produce original complete (compiles and runs on BC4 using batch-mode command and PyTorch) code that replicates the results in the paper. You should produce a report with sections A-F correct and satisfactory. A partially-complete and correct attempt to address sections G, H, I and L is included (i.e. excluding J and K). Any errors or misses do not significantly affect a "replication of results" effort. Replication results (Section H) are within 10% error of expected Avg accuracy or better.

55-64
In addition to the above, sections F, G, H, I and J would be complete, correct and reflective of your understanding of the code and the implementation. All sections (except K) are completed to an acceptable standard. Reported results are within 5% of the expected Avg accuracy or better.

65-70
In addition to the above, a satisfactory attempt to provide improvements (K) on the published results have been achieved, correctly described, with improvements to the results. Marginal improvements will be accepted.

70-75
In addition to the above, the presentation given was to a very good standard with almost no areas of weakness. The proposed improvement is far from random and has been carefully thought of in light of the problem and misclassification errors. Section J should include interesting (rather than random) success and failure cases, with explanations of failure cases. The report's organisation and structure should be very good.

75-80
In addition to the above, the report should be submit-able to a B-class peer review conference, i.e. it shows excellent understanding, correct and complete showcasing of the approach. Statements are concise, and any jargon out of implementation details is avoided. The chosen related work reflects state of the art on this problem. Extensive evidence of analysis, creativity & originality in concise content presentation should be shown. Code is commented, and could be easily understood and re-used by a reader.

80-100
In addition to the above, the produced code and report are exemplary, and could be given as an example for an attempt to replicate this published work. Improvements in results are beyond marginal.

Table 2. [Replicated] Class-wise accuracy of four models with four-layer CNN evaluated on [our split of] UrbanSound8K

Class	LMC (LMCNet)	MC (MCNet)	MLMC	TSCNN
ac	61.0%	81.0%	33.0%	76.0%
ch	100.0%	60.6%	94.0%	97.0%
cp	84.0%	82.0%	86.0%	90.0%
db	80.0%	71.0%	77.0%	78.0%
dr	81.0%	72.0%	85.0%	80.0%
ei	78.5%	75.3%	91.4%	76.3%
gs	93.8%	93.8%	93.8%	93.8%
jh	100.0%	96.9%	100.0%	100.0%
si	65.1%	57.8%	63.9%	62.7%
sm	97.0%	85.0%	92.0%	95.0%
Avg	84.0%	77.5%	81.6%	84.9%