Deep embeddings with Essentia models

Pablo Alonso-Jiménez **Dmitry Bogdanov** **Xavier Serra**

Music Technology Group, Universitat Pompeu Fabra

pablo.alonso@upf.edu

ABSTRACT

We present the integration of various CNN TensorFlow models developed for different MIR tasks into Essentia. This is a continuation of our previous work [1], extending the list of supported models and adding new algorithms to facilitate usability. Essentia provides input feature extraction and inference with TensorFlow models in a single C++ pipeline with Python bindings, facilitating the deployment of C++ and Python MIR applications. We assess the new models' capabilities to serve as embedding extractors in many downstream classification tasks. All presented models are publicly available on the Essentia website.

1. MODELS

We ported the following pre-trained models for a variety of MIR tasks to Essentia, all of them suitable to extract embeddings, which we evaluate in this paper:

- **VGG-I** [2] is a simple VGG model trained for music auto-tagging. We took the output of the penultimate layer for embeddings.
- **VGGish** [3] is a deep VGG model trained to predict tags from Youtube videos. The penultimate layer was specifically designed to produce embeddings.
- **MusiCNN** [2] is another music auto-tagging model with different filter shapes aimed at the music domain. Similarly, we used the output of the penultimate layer.
- **Tempo-CNN** are models for tempo estimation. For embeddings, we selected the DeepSquare [4] model with $k = 16$ and used the logits of the last layer.
- **OpenL3** [5] is a collection of self-supervised models trained to predict coincidence between audio and video chunks. For embeddings, we selected the version with 128 mel bands, trained on musical data.
- **Spleeter** [6] is a collection of source separation models using a separate U-Net architecture for each stem. We selected the 5-stem model and concatenated the bottleneck layers of the stems to create our embedding. We applied 4×4 max-pooling to reduce dimensionality.

Additionally, we trained two variants of the music auto-tagging models, VGG-I-T200 and MusiCNN-T200, using the 200 most frequent tags of MSD-Last.fm [7] instead of just 50. Using more tags allowed to increase the training size from 220K to 350K tracks.

Model	RF (s)	Dims.	Params.	Data size	Appr.
MusiCNN	3	200	787K	220K	FS
MusiCNN-T200	3	200	787K	350K	FS
VGG-I	3	256	605K	220K	FS
VGG-I-T200	3	256	605K	350K	FS
VGGish	1	128	62M	70M	FS
Tempo-CNN	12	256	1.2M	11K	FS
OpenL3	1	512	4.7M	296K	SS
Spleeter	12	1280	49M	-	FS

Table 1. Model embeddings. RF: Receptive field, Approach: fully-supervised (FS) or self-supervised (SS).

Table 1 compares the embeddings in terms of the receptive field, embedding layer dimension, number of parameters of the network and amount of training data.

Dedicated algorithms in Essentia support all the models. The Python code example below shows using VGGish to extract the embeddings: [1]

```
from essentia.standard import MonoLoader,
    TensorflowPredictVGGish

x = MonoLoader(filename='song.wav',
    sampleRate=16000)()

embeddings = TensorflowPredictVGGish(
    graphFilename='audioset-vggish-3.pb',
    output='model/vggish/embeddings')(x)
```

The `output` parameter specifies by name the layer to extract and defaults to the model's main output instead of the embeddings. Extracting other layers requires knowing their name, which can be found with a graph inspection tool such as Netron. [2]

2. DOWNSTREAM TASKS

To evaluate the embeddings, we selected 16 downstream tasks in Table 3, using both public and in-house datasets.

We regarded all the tasks as single-label multiclass classification problems and followed the same evaluation methodology as in our previous study [1]. It includes stratified 5-fold cross-validation and validation on the external MTG-Jamendo dataset containing annotations with the same taxonomies (for most of the tasks) for 10K tracks.

In preprocessing, we removed the problematic tracks in *gtzan* [8]. For *fs-loop-ds* [9], we only used tracks annotated

[1] More examples available at https://essentia.upf.edu/machine_learning.html

[2] https://github.com/lutzroeder/netron

Task	MusiCNN			MusiCNN-T200			VGG-I			VGG-I-T200			VGGish			OpenL3			Spleeter			Tempo-CNN		
	5F	JD	AD	5F	JD	AD	5F	JD	AD	5F	JD	AD	5F	JD	AD	5F	JD	AD	5F	JD	AD	5F	JD	AD
dortmund	61	46	15	46	41	5	54	12	42	26	22	4	50	48	2	38	21	17	35	24	11	16	17	-1
gtzan	86	54	32	79	47	32	83	53	30	46	26	20	84	62	22	58	14	44	57	32	25	26	15	11
rosamerica	94	58	36	90	60	30	93	59	34	66	32	34	93	59	34	84	24	60	70	33	37	46	33	13
voice/instrum.	98	83	15	93	82	11	97	79	18	78	71	7	98	87	11	89	54	35	76	65	11	58	55	3
tonal/atonal	87	60	27	91	61	30	92	61	31	78	55	23	93	64	29	89	51	38	89	60	29	70	59	11
gender	87	82	5	79	76	3	84	80	4	70	65	5	83	79	4	55	53	2	55	62	-7	51	53	-2
danceability	98	66	32	94	70	24	94	68	26	71	62	9	94	70	24	90	58	32	90	62	28	66	59	7
acoustic	96	70	26	93	74	19	93	73	20	83	64	19	93	74	19	89	55	34	89	62	27	75	61	14
aggressive	97	72	25	97	76	21	99	67	32	82	70	12	99	67	32	91	52	39	93	58	35	69	59	10
electronic	93	78	15	88	77	11	88	76	12	74	70	4	94	81	13	77	57	20	77	63	14	64	55	9
happy	86	57	29	77	55	22	89	62	27	69	58	11	86	60	26	76	51	25	70	55	15	68	57	11
party	92	77	15	92	75	17	64	68	-4	84	73	11	90	75	15	77	57	20	87	66	21	73	63	10
relaxed	89	71	18	86	67	19	91	71	20	79	65	14	90	71	19	81	53	28	80	61	19	72	60	12
sad	87	67	20	88	65	23	86	68	18	83	62	21	89	65	24	85	55	30	83	60	23	84	62	22
fs-loop-ds	56	-	-	49	-	-	53	-	-	38	-	-	53	-	-	53	-	-	46	-	-	24	-	-
urbansound8k	81	-	-	40	-	-	82	-	-	35	-	-	89	-	-	77	-	-	70	-	-	10	-	-

Table 2. Class-weighted accuracies (%) in the downstream tasks for the embeddings. 5F: 5-fold cross-validation. JD: validation on the MTG-Jamendo dataset. AD: accuracy drop (5F - JD). Top 5F/JD results are shaded light/medium gray.

by a single class and merged "melody" and "voice" classes to compensate for class unbalance.

For each task, we trained a classifier on top of the different embeddings produced by our models. The architecture consists of a multilayer perceptron with a single hidden layer with 100 neurons and ReLU activations. The output layer uses softmax for predictions.

Table 2 reports the results in both evaluations. The embeddings produced by the MusiCNN, MusiCNN-T200, VGG-I, and VGGish models achieved the best performance in different tasks. We observe different generalization abilities according to the accuracy drop between cross-validation and the performance on previously unseen data.

3. USES IN MIR

Our final goal is to provide fast C++ inference for state-of-the-art deep learning models in Essentia suitable for deployment in diverse MIR applications. To this end, we offer models for specific use-cases (auto-tagging, tempo estimation, source separation, and music classification by genre, mood, and instrumentation).

Additionally, our algorithms are designed in a way that allow using the models as feature extractors. Some of them provide embeddings suitable for transfer learning. All models created in this study are available online. [3]

	Dataset	Classes	Size
genre	dortmund	alternative, blues, electronic, folk-country, funksoulrnb, jazz, pop, raphiphop, rock	1820 exc.
	gtzan	blues, classic, country, disco, hip hop, jazz, metal, pop, reggae, rock	1000 exc.
	rosamerica	classic, dance, hip hop, jazz, pop, rhythm and blues, rock, speech	400 ft.
mood	acoustic	acoustic, non acoustic	321 ft.
	aggressive	aggressive, non aggressive	280 ft./exc.
	electronic	electronic, non electronic	332 ft./exc.
	happy	happy, non happy	302 exc.
	party	party, non party	349 exc.
	relaxed	relaxed, non relaxed	446 ft./exc.
	sad	sad, non sad	230 ft./exc.
miscellaneous	danceability	danceable, non danceable	306 ft.
	voice/instrum.	voice, instrumental	1000 exc.
	gender	male, female	3311 ft.
	tonal/atonal	atonal, tonal	345 exc.
	urbansound8k	air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shot, jackhammer, siren, street music	8732 exc.
	fs-loop-ds	bass, chords, fx, melody, percussion	2104 exc.

Table 3. Downstream tasks (ft.: full tracks, exc.: excerpts).

[3] https://essentia.upf.edu/models/

4. REFERENCES

[1] P. Alonso-Jiménez, D. Bogdanov, J. Pons, and X. Serra, "Tensorflow audio models in essentia," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'20)*, 2020.

[2] J. Pons and X. Serra, "musicnn: Pre-trained convolutional neural networks for music audio tagging," in *International Society for Music Information Retrieval Conference (ISMIR'19) LBD*, 2019.

[3] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold *et al.*, "CNN architectures for large-scale audio classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'17)*, 2017.

[4] H. Schreiber and M. Müller, "Musical tempo and key estimation using convolutional neural networks with directional filters," in *Sound and Music Computing Conference (SMC'19)*, 2019.

[5] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen and learn more: Design choices for deep audio embeddings," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'19)*, 2019.

[6] R. Hennequin, A. Khlif, F. Voituret, and M. Moussallam, "Spleeter: a fast and efficient music source separation tool with pre-trained models," *Journal of Open Source Software*, vol. 5, no. 50, p. 2154, 2020.

[7] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *International Society for Music Information Retrieval Conference (ISMIR'11)*, 2011.

[8] B. L. Sturm, "The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use," *arXiv preprint:1306.1461*, 2013.

[9] A. Ramires, F. Font, D. Bogdanov, J. B. Smith, Y.-H. Yang, J. Ching, B.-Y. Chen, Y.-K. Wu, H. Wei-Han, and X. Serra, "The Freesound Loop Dataset and annotation tool," in *International Society for Music Information Retrieval Conference (ISMIR'20)*.