CountNet: Estimating the Number of Concurrent Speakers Using Supervised Learning

Fabian-Robert Stöter, Soumitro Chakrabarty, Bernd Edler, Emanuël A. P. Habets

CountNet: Estimating the Number of Concurrent Speakers Using Supervised Learning

Fabian-Robert Stöter
International Audio Laboratories Erlangen*

Soumitro Chakrabarty
International Audio Laboratories Erlangen

Bernd Edler
International Audio Laboratories Erlangen

Emanuël A. P. Habets
International Audio Laboratories Erlangen

Abstract

Estimating the maximum number of concurrent speakers from single-channel mixtures is a challenging problem and an essential first step to address various audio-based tasks such as blind source separation, speaker diarization, and audio surveillance. We propose a unifying probabilistic paradigm, where deep neural network architectures are used to infer output posterior distributions. These probabilities are in turn processed to yield discrete point estimates. Designing such architectures often involves two important and complementary aspects that we investigate and discuss. First, we study how recent advances in deep architectures may be exploited for the task of speaker count estimation. In particular, we show that convolutional recurrent neural networks outperform recurrent networks used in a previous study when adequate input features are used. Even for short segments of speech mixtures, we can estimate up to five speakers, with a significantly lower error than other methods. Second, through comprehensive evaluation, we compare the best-performing method to several baselines, as well as the influence of gain variations, different datasets, and reverberation. The output of our proposed method is compared to human performance. Finally, we give insights into the strategy used by our proposed method.

1 Introduction

In a "cocktail-party" scenario, one or more microphones capture the signal from many concurrent speakers. In this setting, different applications may be envisioned such as localization, crowd monitoring, surveillance, speech recognition, speaker separation, etc. When devising a system for such a task, it is typically assumed that the actual number of concurrent speakers is known. This assumption turns out to be of paramount importance for the effectiveness of subsequent processing. Notably, for separation algorithms [57], real-world systems do not straightforwardly provide information about the actual number of concurrent speakers. It, therefore, is desirable to close the gap between theory and practice by devising reliable methods to estimate the number of sound sources in realistic environments. Surprisingly, very few methods exist for this purpose in an audio context, in particular from a single microphone recording.

From a theoretical perspective, estimating the number of concurrent speakers is closely related to the more difficult problem of *identifying* them, which is the topic of speaker diarization [6, 60, 62, 63]. Intuitively, if a system is able to tell who speaks when, it is naturally also able to tell how many speakers are actually active in a mixture. We call this strategy "counting by detection". A good working diarization system would be able to sufficiently address the speaker count estimation

*International Audio Laboratories Erlangen is a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institute for Integrated Circuits (IIS).

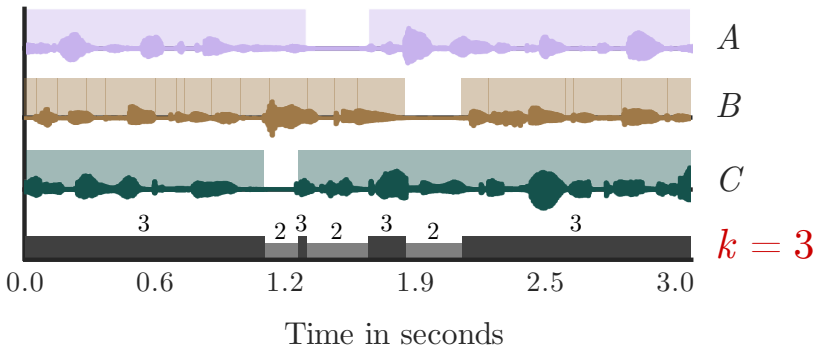


Figure 1: Illustration of our application scenario of three concurrent speakers (A, B, C) and their respective speech activity. Bottom plot shows the mixture (input), the number of concurrently active speakers and its maximum k which is our targeted output.

problem using this strategy. However, it appears to be a very complex problem to tackle when one is only interested in the number of concurrent speakers. Furthermore, as current diarization systems only work when a clear segmentation is possible, the first step of such a system often is to find homogeneous segments in the audio where only one speaker is active. The segment borders can be found by speaker change detection [84]. These homogeneous segments are used to discriminate and temporally locate the speakers within a given recording. When sources are simultaneously active, as in real cocktail party environments, existing segmentation strategies fail. In fact, overlapping speech segments typically are a major source of error in speaker diarization [6].

To improve the robustness of these detection-based methods, a number of approaches attempt to detect and possibly reject the overlapping speech segments to improve performance [13,36]. Overlap detection has since evolved into its own line of research with many recent publications such as [4, 26, 30, 71]. Overlap detection can be seen as a binarized version of the count estimation problem where the number of speakers equals to one (*no overlap*) or more than one (*overlap*). It is, therefore, possible to apply a count estimation system for the overlap detection problem but not vice versa. Also, an overlap detection system cannot be easily utilized in a source separation system. In fact, it should be noted that before the arrival of deep learning based separation systems, models required long context and in such a case, for methods like NMF, the number of concurrent speakers could be introduced as a regularization term [45]. In recent years, however, large improvements were achieved by deep learning based methods [31, 85] at shorter segment duration (often 1-5 seconds). In such approaches, it becomes possible to apply separation only when its "needed". In this scenario, a method of estimating the maximum number of concurrent speakers becomes useful and in some cases essential.

When speaker overlap is as prevalent as in a "cocktail-party" scenario, developing an algorithm to detect the number of speakers is challenging. This is in contrast to humans whom we know are excellent in segregating one source from a mixture [15] and tend to use this skill to perceptually segregate speakers before they can estimate a count, as highlighted, e.g. in [42]. As shown in [41, 42], humans are able to correctly estimate up to three simultaneously active speakers without using spatial information. Similarly, in music, psycho-acoustic researchers came up with a "one-two-three-many" hypothesis [37, 68, 75]. The question if machines could outperform humans, or if they are subject to similar limitations, remains to be answered.

Identifying isolated sources in realistic mixtures is challenging [15] and psychology studies in vision [39] have shown that humans can instantly estimate the number of objects without actually counting and therefore identifying them. This phenomenon is known as *subitizing* and has been inspiring research in vision [17]. Since there are indications that the auditory system is also capable of subitizing sources [77], we transfer this fact to the audio domain and directly attempt in this study to estimate the number of audio sources instead of counting them after identification. We refer to this strategy as "direct count estimation".

Directly estimating the number of sources in audio mixtures has many applications and appears as a reasonable objective that mimics the process of human perception. Since humans do have two ears that provide spatial diversity, a first natural idea to imitate human performance is to exploit *binaural* information to proceed to source count estimation. In terms of signal processing, this is achieved by estimating directions of arrival (DoA) and clustering them [8,9,22,47,52,55,56,79]. However, many audio devices are equipped with only a single microphone, and being able to also count sources, in that case, is desirable. Consequently, the single-channel scenario has been considered in many studies.

One of the first single-channel methods was proposed in 2003 by Arai [7]. It is based on the assumption that speech mixed from more than one speaker has a more complex amplitude modulation pattern than a single speaker. The modulation pattern is aggregated and used as a decision function to distinguish between different numbers of speakers. In [65], the authors propose an energy feature based on temporally averaged mel filter outputs. The number of concurrent speakers was determined by manually determining thresholds that best match individual speaker counts. In a more recent work, Xu et.al. [82] estimate the number of speakers by applying hierarchical clustering to fixed-length audio segments using mel frequency cepstral coefficients (MFCCs) and additional pitch features. The method assumes the presence of at least some non-overlapped speech and was evaluated on real-world data of 20 hours duration. An average count estimation error of one speaker is reported using excerpts of eight-minutes duration and featuring up to eight speakers. In another vein, Andrei et.al. [5] proposed an algorithm which correlates single frames of multi-speaker mixtures with a set of single-speaker utterances. The resulting score was then used to estimate the number of speakers using thresholds.

In all the aforementioned methods, the speaker count estimation problem was devised. The different strategies undertaken there rely on classical and grounded signal processing strategies and exhibit fair performance in a controlled setup. However, our experience shows (see Section 6) that they leave much room for improvement when applied to more diverse and challenging signals than those corresponding to their targeted applications, notably in the case of many different and constantly overlapping speakers. This is due to their main common weakness, which is to rely on the assumption that there are segments where only one speaker is active, in a way that is similar to the classical speaker diarization studies mentioned before. In [76] a first data-driven approach based on a recurrent network was presented, motivated by the recent and impressive successes of deep learning approaches in various audio tasks like speech separation [27, 31, 85] and speaker diarization [24, 35, 83]. The methods proposed in [76] to address speaker count estimation using deep learning were built upon recent methods to count objects in images, which is a popular application with many contributions from the deep learning community [10, 14, 17, 43, 48, 69, 80, 86, 87]. In [76] two main paradigms were evaluated: a) count estimation as regression problem, where the systems are directly trained to output the number of objects as a point estimate, and b) classification, where every possible number of objects is encoded as a different class and the output of a predicting system corresponds to a probability distribution over these classes. The results of the proposed method indicated that a classification based neural network performed better than one based on regression. One drawback, however, is that the maximum number of speakers (the number of classes) is known in advance.

In this study, we build upon [76] and focus on the network architecture design, as well as on finding limitations for different test scenarios. This work makes the following contributions: i) we generalize the problem formulation by fusing classification and regression, which allows estimating discrete outputs while controlling the error term. This is done by picking a point estimate from a full posterior distribution provided by the deep architectures; ii) in addition to the recurrent network introduced in [76], we propose alternative speaker-independent neural network architectures based on the convolution operation to improve count estimation. Each of the proposed networks is adjusted to estimate the number of speakers from audio segments of 5 seconds; iii) we test the performance of these networks in multiple experiments and compare them to several baseline methods, pointing out possible limitations. Furthermore, we present a statistical analysis of the results to determine whether classification outperforms regression for all architectures; iv) we conducted a listening experiment to relate the best-performing machine to human performance. We describe one of the strategies taken by the data-driven approach that might explain its superior performance.

Finally, for the sake of reproducibility, the trained networks (models), as well as the test dataset, are made available on the accompanying website[2] and the github repository[3].

The remainder of this paper is organized as follows. In Section 2, we describe the count estimation problem formally and the general ideas we propose to tackle it. In Section 3, we propose several architectures, each of them adjusted to estimate the number of speakers from short audio segments of 5 s. In Section 5, we then assess several common hyperparameters for all of our proposed architectures, so that we are able to propose a single, best-performing model. In Section 6 this model is compared to several baseline systems under various acoustical conditions. Additionally, we compare the proposed method to human performance. We point out possible limitations and provide indications for the strategy being taken by the DNN in Section 7 before we conclude in Section 8.

2 Problem Formulation

We consider the task of estimating the maximum number of concurrent speakers $k \in \mathbb{Z}_0^+$ in a single-channel audio mixture $\mathbf{x}$. This is achieved by applying a mapping from $\mathbf{x}$ to k. We now provide details on the notations, the general structure of the method, and various ways to exploit the deep learning framework to estimate k.

2.1 Signal Model

Let $\mathbf{x}$ be a time domain vector with N samples, representing a linear mixture of L single speaker speech signal vectors $\mathbf{s}_l$. The value observed at time instant n for the mixture is given by x_n and for the individual speech segments by s_{nl}. The mixture then results in

$$x_n = \sum_{l=1}^{L} s_{nl} \; \forall n \in \mathbb{Z}^N. \tag{1}$$

Naturally, each speaker $l = 1, \ldots, L$ is not active at every time instant. On the contrary, we assume there is a latent binary *speech activity* variable $v_{nl} \in \{0, 1\}$ that is either provided by a ground truth annotation or computed using a voice activity detection method.

Our objective of estimating the maximum number of concurrent speakers can now be formulated as

$$k = \max_n \left(\sum_{l=1}^{L} v_{nl} \right) \; n \in \{1, \ldots, N\}. \tag{2}$$

As can be seen, our proposed task of estimating $k \leq L$, is more closely related to source separation whereas the estimation of L is more useful for tasks where speakers do not overlap. For instance, three non-overlapping speakers would result in $L = 3$ and $k = 1$. It should be noted that at short time scales both task definitions provide the same outcome because on such a time scale the speaker configuration usually does not change. The problem arises for long-term recordings (e.g. larger than ten seconds) which are not considered in this work. In any case, we want to emphasize that in all experiments presented in this paper, we made sure that for all audio segments $L = k$.

In the remainder of this work, we assume that no additional prior information about the speakers is given to the system except possibly the maximum number of concurrent speakers $k_{\max}$, that is application-dependent and represents an upper bound for the estimation.

While speaker diarization would mean estimating the whole speech activity matrix v_{nl}, our problem of estimating only k in (2) is more abstract as it requires a direct estimation of the count as advocated in Section 1.

In Figure 1, we illustrate our setup in a "cocktail-party" scenario featuring $L = 3$ unique speakers. At any given time, we see that at most $k = L = 3$ speakers are active at the same time and $k = 2$ could be the outcome if a smaller excerpt would be evaluated. By processing such excerpts in a sliding-window fashion, our proposed solution can be applied straightforwardly to context sizes commonly used in source separation. Furthermore, our proposed system can be used also to detect overlap ($k > 1$), which can be useful as a pre-processing step for diarization.

[2] `https://www.audiolabs-erlangen.de/resources/2017-CountNet`.

[3] `https://github.com/faroit/CountNet`.

Now, the system we propose is actually not inputting the signal vector $\mathbf{x}$, but rather a Time-Frequency (TF) representation as the absolute value of the short-time Fourier transform of $\mathbf{x}$ that is denoted by $\mathbf{X}$. In the following, $\mathbf{X}$ is the non-negative input for the system.

2.2 Probabilistic Formulation

In a supervised scenario, let $\{\mathbf{X}_t, k_t\}_t$ be all of our learning examples, where $t \in 1, \ldots, T$ denotes the t-th training item from the training database. For the purpose of learning a mapping between $\mathbf{X}$ and k, we adopt a probabilistic viewpoint and introduce a flexible generative model that explains how a particular source count k corresponds to some given input $\mathbf{X}$.

First, we consider that all training samples $\{\mathbf{X}_t, k_t\}_t$ are independent. For each sample, we consider that k_t is drawn from a probability distribution of a known parametric family, parameterized by some latent and unobserved parameters $\mathbf{y}_t$

$$\mathbb{P}\left(k_t \mid \mathbf{X}_t\right) = \mathcal{L}\left(k_t \mid \mathbf{y}_t\right), \tag{3}$$

the distribution $\mathcal{L}\left(\cdot \mid \mathbf{y}_t\right)$ is called the *output distribution* in the following. We further assume that there is some deterministic mapping between $\mathbf{X}_t$ and $\mathbf{y}_t$, embodied as

$$\mathbf{y}_t = f_\theta\left(\mathbf{X}_t\right), \tag{4}$$

where θ are the parameters for this deterministic mapping, that is independent of the training item t. This results in an output distribution given by

$$\mathbb{P}\left(k_t \mid \mathbf{X}_t\right) = \mathcal{L}\left(k_t \mid f_\theta\left(\mathbf{X}_t\right)\right). \tag{5}$$

Assume for the rest of this section that these parameters θ are known. Given a previously unseen input $\mathbf{X}$, expression (5) means we can compute the distribution of the source count k.

The objective of our counting system is to produce a point estimate $\hat{k}$ rather than a whole output distribution $\mathbb{P}\left(k \mid \mathbf{X}\right)$. A first option is to pick as an estimate the most likely outcome for the output distribution, thus resorting to Maximum A Posteriori (MAP) estimation:

$$\hat{k} = \underset{k}{\operatorname{argmax}}\ \mathcal{L}\left(k \mid f_\theta\left(\mathbf{X}\right)\right). \tag{6}$$

However, MAP is not the only option and a broad range of point estimation techniques may be obtained when resorting to decision theory [11]. We may for example also choose $\hat{k}$ as the value that minimizes the marginal average cost of choosing an estimate $\hat{k}$ instead of the true value k, when k is distributed with respect to the output distribution

$$\hat{k} = \underset{u}{\operatorname{argmin}} \int_k d\left(k, u\right) \mathcal{L}\left(k \mid f_\theta\left(\boldsymbol{X}\right)\right) \mathrm{d}k, \tag{7}$$

where $d\left(k, u\right)$ is the cost of picking u as an estimate when the true value is k. It may be any function that seems appropriate, and does not necessarily need to be differentiable. However, we retain the more general formulation (7) because other choices will sometimes prove more effective, as we show later. For notational convenience, we write (7) as

$$\hat{k} = q\left(f_\theta\left(\boldsymbol{X}\right)\right), \tag{8}$$

and $q\left(\cdot\right)$ is called the *decision function*. Using this strategy, we have everything to produce a single source count estimate $\hat{k}$ from input features $\mathbf{X}$, provided the parametric family $\mathcal{L}$ and the mapping f_θ as well as its parameters θ are known.

In this study, we choose a deep neural network for the mapping f_θ, whose weights θ are trained in a supervised manner. Once a particular network architecture has been chosen, learning its parameters is achieved through classical stochastic gradient descent. If we assume that the particular family $\mathcal{L}$ of output distributions has been chosen, it appears natural to learn the parameters θ that maximize the likelihood of the learning data. More specifically, the total cost to be minimized becomes

$$C = \sum_{t=1}^{T} -\log \mathcal{L}\left(k_t \mid f_\theta\left(\boldsymbol{X}_t\right)\right). \tag{9}$$

The derivative of this cost (9) with respect to the parameters can be used to learn the network parameters.

Three different choices for the family of output distributions (classification, Gaussian regression and Poisson regression) are summarized below.

2.2.1 Classification

In a classification setting, the output distribution is directly taken as *discrete*, discarding any meaning concerning the ordering of the different possible values. Given some particular input $\mathbf{X}$, the network generates the posterior output probability for $(k_{\max}+1)$ classes (including $k=0$) and a maximum a posteriori (MAP) decision function is chosen that simply picks the most likely class $q = \arg\max(\cdot)$.

Classification based approaches have successfully been applied in deep neural networks for counting objects [43, 69, 87] in images.

2.2.2 Gaussian Regression

In regression, k is derived from an output distribution defined on the real line. However, this comes with the additional difficulty of handling the fact that k is integer.

The output distribution in this setting is assumed to be Gaussian and the associated cost function is the classical squared error. During inference and given the output $f_\theta(\mathbf{X})$ of the network, the best discrete value that is consistent with the model is simply the rounding operator $q = [\cdot]$.

Gaussian regression has achieved state-of-the-art counting performance in computer vision using deep learning frameworks [14, 48, 86].

2.2.3 Discrete Poisson modeling

When it comes to modeling count data, it is often shown effective to adopt the Poisson distribution [23]. First, this strategy retains the advantage of the classification approach to directly pick a probabilistic model over the actual discrete observations, avoiding the somewhat artificial trick of introducing a latent variable that would be rounded to yield the observation. Second, the model avoids the inconvenience of the classification approach to completely drop dependencies between classes.

Due to these advantages, the Poisson distribution has been used in studies devising deep architectures for counting systems [61]. For instance in [16, 23, 61], it is shown that the number of objects in images can be well modeled by the Poisson distribution. Inspired by these previous works, we also consider the Poisson output distribution $\mathcal{P}(k \mid f_\theta(\mathbf{X}))$ where $\mathcal{P}(\cdot \mid \lambda)$ denotes the Poisson distribution with scale parameter λ.

In that setup, the cost function at learning time is the Poisson negative log-likelihood and the deep architecture at test time provides the predicted scale parameter $f_\theta(\mathbf{X}) \in \mathbb{R}_+$, which summarizes the whole output distribution.

As a decision function q in this setting, we considered several alternatives. A first option is to again resort to MAP estimation and pick the mode $[f_\theta(\mathbf{X})]$ of the distribution as a point estimate. However, experiments showed that the posterior median yields better estimates, and is given by

$$q(f_\theta(\mathbf{X})) = \operatorname*{argmin}_{\hat{k}} \sum_{k=0}^{\infty} \left|\hat{k} - k\right| \mathcal{P}(k \mid f_\theta(\mathbf{X})) \tag{10a}$$

$$= \text{median}\left(k \sim \mathcal{P}(f_\theta(\mathbf{X}))\right) \tag{10b}$$

$$\approx \left\lfloor f_\theta(\mathbf{X}) + \frac{1}{3} - \frac{0.02}{f_\theta(\mathbf{X})} \right\rfloor, \tag{10c}$$

where the last expression is an approximation of the median of a Poisson distributed random variable of scale parameter $f_\theta(\mathbf{X})$ [19].

3 DNNs for Count Estimation

Applying deep learning to an existing task often is a matter of choosing a suitable network architecture. Typically an architecture describes the overall structure of the network including (but not limited to) the type and number of layers in the network and how these layers are connected to each other. In turn, designing such an architecture requires deep knowledge about input and output representations and their required level of abstraction. Many audio-related applications like speech

recognition [32] or speaker diarization share similar common architectural structures, often found by incorporating domain knowledge and through extensive hyper-parameter searches. For our task of source count estimation, however, domain knowledge is difficult to incorporate, as our studies aim at revealing the best strategy to address the problem. This is why we chose architectures that already have shown a good level of generalizability for audio applications.

3.1 Network Architectures

The input of all networks is a batch of samples, represented as time-frequency representations $\mathbf{X} \in \mathbb{R}^{D \times F \times C}$, where D refers to the time dimension, F to the frequency dimension and C to the channel dimension (in the single-channel case, $C = 1$). In the following, we discuss several commonly used DNN architectures and their benefits in using them for the task of estimating the number of speakers. All architectures under investigation are summarized in Fig. 2.

3.1.1 Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) are a variant of standard fully-connected neural networks, where the architecture generally consists of one or more "convolution layers" followed by fully-connected layers leading to the output.

Since the individual elements of the filters (weights) are learned during the training stage, convolutional layers can also be interpreted as feature extractors. By stacking up additional layers, CNNs can extract more abstract features in higher level layers [73].

The sizes of the filter kernels are crucial, and it was shown in [59] that many audio applications can benefit if domain knowledge is put into the design of the filter kernel size. The use of small filter kernels, as often used in image classification tasks, does not necessarily decrease performance, when combined with many layers. Also larger kernels increase the number of parameters and therefore the computational complexity. It was shown in [67] that 3×3 kernels resulted in state-of-the-art results in singing voice detection tasks. Due to its hierarchical architecture, CNNs with small filters have the benefit that they can model time and frequency invariances regardless of the scaling of the frequency axis.

Our proposed architecture is similar to the ones proposed by [66] used for singing voice activity detection. In our proposed CNN, we consider local filters of size 3×3. In the first layer, 2D convolution is performed by moving the filter across both dimensions of the input in steps of 1 element (striding $s = 1$ to generate $C = 64$ feature maps/channels resulting in an output volume of $64 \times (D - 3 + 1) \times (F - 3 + 1)$. In the subsequent convolution layers, a similar operation is applied but for each convolutional layer, we consider a different number of feature maps. Note, that the convolution operation is performed independently for every input channel, and then summed up along the dimension C for each output element. In preliminary experiments we found that by using max-pooling we received significantly better performance when used after CNN layers.

3.1.2 Recurrent Neural Networks (RNN)

While convolutional layers excel in capturing local structures, RNNs can detect structure in sequential data of arbitrary length. This makes it ideal to model time series; however, in practice, the learned temporal context is limited to only a few time instances, because of the vanishing gradient problem [33]. To alleviate this problem, forgetting factors (also called gating) were proposed. One of the most popular variants of RNNs with forgetting factors is the Long Short-Term Memory (LSTM) [34] cell. In [76] such an architecture based on three bi-directional LSTM cells, was proposed. The architecture is similar to the one employed in [46].

3.1.3 Convolutional Recurrent Neural Networks (CRNN)

Recently, a combination of convolutional and recurrent layers were proposed for audio-related tasks [3, 18, 64, 88].

The main motivation to stack these layers is to combine the benefits of convolutional layers with those of recurrent architectures, namely the benefit of convolutional layers in aggregating local features with the ability of recurrent layers to model long-term temporal data.

Table 1: Parameter Optimization of F-CNN Model through hyper-parameter search. Bold hyper-parameters were found optimal.

Layer	Parameters	Value Range
CNN 1	Feature Maps	$\{16, \mathbf{32}, 64\}$
CNN 1	Filter Length	$\{\mathbf{3}, 5, 7\}$
Pooling 1	Pooling Length	$\{1, \mathbf{2}, 4\}$
CNN2	Feature Maps	$\{16, \mathbf{32}, 64\}$
CNN2	Filter Length	$\{\mathbf{3}, 5, 7\}$
Pooling 2	Pooling Length	$\{1, 2, 4\}$
CNN 3	Presence of Layer	$\{\mathbf{Yes}, No\}$
CNN 3	Feature Maps	$\{16, 32, \mathbf{64}, 128\}$
CNN 3	Filter Length	$\{\mathbf{3}, 5, 7\}$
Pooling 3	Pooling Length	$\{1, \mathbf{2}, 4\}$
Fully Connected 1	Hidden Unit	$\{64, \mathbf{128}\}$
Dropout 1	Dropout Percentage	$[0.1, \mathbf{0.2}, 0.5]$
Fully Connected 2	Hidden Unit	$\{32, \mathbf{48}\}$
Dropout 2	Dropout Percentage	$[0.1, \mathbf{0.2}, 0.5]$

There are different ways to stack CNNs and RNNs to form a CRNN architecture. In our application the motivation is to aggregate local time-frequency features coming from the output convolutional neural network and use the LSTM layer to model long temporal structures. As the output of a CNN layer is a 3D volume $D \times F \times C$ and the input of a recurrent layer only takes a 2D sequence, the dimension would need to be reduced. Naturally, the time dimension would need to be kept, therefore the channel dimension C is stacked with the frequency dimension F resulting in a $D \times F \cdot C$ output.

3.1.4 Full-band Convolutional Neural Networks (F-CNN)

Architectures where filters span the full frequency range and therefore apply convolution in temporal direction only, have already been successfully deployed in speech [3] and music application [18, 21, 58]). Our motivation here is that the activity of speakers happen over wide frequency ranges and a count (unlike in counting objects in images) cannot be split into sub counts. The full-band kernel configuration only affects the first hidden layer, as in consecutive outputs all frequency bands are squashed down to one single frequency band using "valid" convolutions. This is computationally very efficient, because it reduces the middle layer's dimensionality of the network significantly due to this aggregation. To further optimize the performance of the network, we applied a hyper-parameter optimization technique using Tree-structured Parzen Estimator (TPE) [12]. We used a search space of several hyper-parameters as shown in Table 1 and set the maximum number of evaluations to 200.

The results are in agreement with the findings in [66] where small filter kernels of size 3 outperformed larger kernels. Also, it can be seen from the results, that increasing the number of feature maps of the convolutional layers does not necessarily increase the performance.

3.1.5 Full-Band Convolutional Recurrent Neural Networks (F-CRNN)

Similarly to *CRNN* and to the Deep Speech 2 implementation [3], we added an LSTM recurrent layer to the output of the last convolutional layer. Since each filter output is only of dimension one, an additional flattening as in *CRNN* is not required.

3.2 Output Activation Functions for Count Estimation

For each of the decision functions a suitable output activation and loss is used.

3.2.1 Classification

For *classification*, the output is required to be one-hot-encoded so that the output is of dimension $y \in \mathbb{B}^{L+1}$, where L is the maximum number of concurrent speakers to be expected. In the final

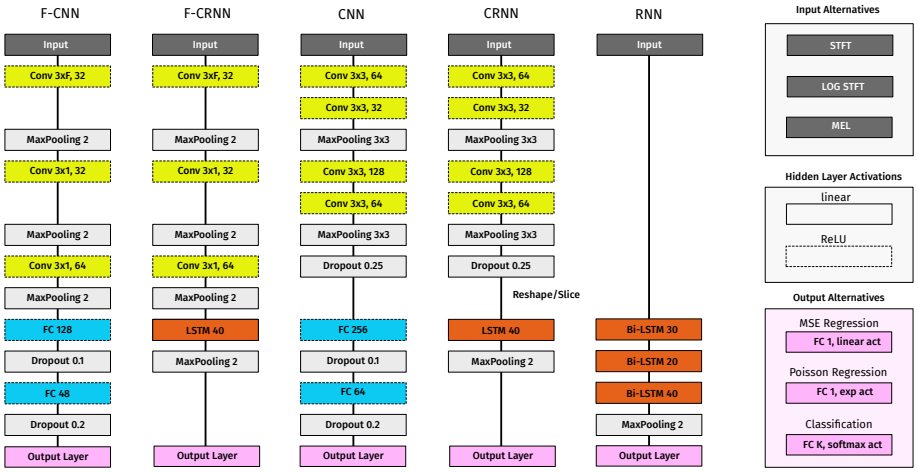


Figure 2: Overview of the proposed architectures.

layer of the network, a softmax activation function is used with the cross-entropy function as the loss.

3.2.2 Gaussian Regression

For the Gaussian regression model, the final output layer is of dimension $y \in \mathbb{R}^1$. The output layer nodes have linear activation, and mean squared error is used as the loss function.

3.2.3 Poisson Regression

For the Poisson regression, the likelihood of parameter λ given the true count k is computed by the negative log-likelihood loss $E = \sum \lambda - k * \log(\lambda + eps)$. The output layer activation is the exponential function.

4 Training

To successfully train and evaluate the proposed DNNs, due to the number of parameters, a large amount of training data is required. In this section, we introduce relevant speech corpora and describe how the training dataset was assembled.

4.1 Speech Corpora and Annotations

To date, many available speech datasets contain recordings where only a single speaker is active. Datasets that include overlapped speech segments, either lack accurate annotations because the annotation of speech onsets and offsets in mixtures is cumbersome for humans as shown in Section 1 or lack a controlled auditory environment such as in TV/broadcasting scenarios [28]. Since a realistic dataset of fully overlapped speakers is not available, we chose to generate synthetic mixtures. We recognize that in a simulated "cocktail-party" environment, mixtures lack the conversational aspect of human communication but provide a controlled environment which helps to understand how a DNN solves the count estimation problem. As we aim for a speaker independent solution, we selected a speech corpus with preference to a high number of different speakers instead of the number of utterances, thus increasing the number of unique mixtures. We selected *LibriSpeech clean-360* [54] which includes 363 hours of clean speech of English utterances from 921 speakers (439 female and 482 male speakers) sampled at 16 kHz. In the further course of this work (see Section 6), we also present the results from test sets of two other datasets as listed in Table 2. Furthermore, we included non-speech examples from the TUT Acoustic Scenes dataset [51] in our training data to avoid using zero input samples for $k = 0$ to increase the robustness against noise.

Table 2: Overview of speech corpora used in this work.

		Number of Speakers		
Name	Language	Train	Valid.	Test
LibriSpeech [54]	English	921	40	40
TIMIT [25]	English	462	24	168
THCHS [81]	Mandarin	30	10	10

A single training tuple $\{\mathbf{X}, k\}$ is generated by a synthetic speech mixture and their ground truth speaker count k. The mixtures are formed from random utterances of different speakers where silence in the beginning and end was removed to increase the overlap within one segment. In fact, our method to generate synthetic samples results in an average overlap for $k = 2$ of 85% and for $k = 10$ of 55% (based on 5s segments). This procedure is similar to [50] used to label the data. Signals are mixed according to (1), peak normalized and then transformed to a time-frequency matrix $X \in D \times F$. Based on a voice activity detection algorithm (VAD), we used an implementation based on the WebRTC Standard [1] where we computed the ground truth output k via (2). All samples are normalized to the average Euclidean norm of $duration$ frames to be robust against gain variations as proposed by [78]. Furthermore, the data was scaled to zero mean and unit standard deviation across the frequency dimension F over the full training data. Scaling parameters were saved for validation and test. For a more detailed description of the data set, the reader is referred to [76].

4.2 Training Procedure

For all experiments we chose a medium sized training dataset of $k \in \{0, \ldots, 10\}$ forming a total of $T_{\text{train}} = 20.020$ mixtures (1820 per k), each containing 10 seconds of audio, resulting in 55.55 hours of training material. For each sample fed into the network, we select a random excerpt of duration D from each mixture. If not stated otherwise, $D = 5$ seconds. That way, for each epoch, the network is seeing slightly different samples, reducing the number of redundant samples and thus helping to speed up the stochastic gradient based training process. [4] A similar training procedure is detailed in [66, 76]. Each architecture is trained using the ADAM optimizer [44] (learning rate: $1 \cdot 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \cdot 10^{-8}$) using mini-batches of size 32. Our training procedure verifies that all samples within a batch are from a different set of speakers. In addition to the training dataset, we created a fully separated validation dataset of $T_{\text{valid}} = 5720$ samples using a different set of speakers from *LibriSpeech dev-clean*. Early stopping ($patience = 10$) is applied by monitoring the validation loss to reduce the effect of overfitting. Training never exceeded more than 50 epochs.

We used the Keras [20] framework and trained on multiple instances of Nvidia GTX 1080 GPUs.

5 Model Selection

In this section, we evaluate three configurations of our proposed architectures, introduced in Section 3. Besides the architecture, we investigate different input representations as well as the three proposed output distributions (see Section 2). The goal of this is to determine the effect of these parameters and fix them to select a final trained network (model) based on these parameters.

To allow for a controlled test environment and at the same time limit the number of training iterations, we fix certain parameters: In this experiment, the level of the speakers was adjusted before mixing such that they have equal power. Furthermore, the input duration D was fixed to five seconds. For all experimental parameters, we repeated the training three times with different random seeds for each run and report averaged results to minimize random effects caused by early stopping. We used the *LibriSpeech* dataset for both training and validation and performed evaluation of all models on $T_{\text{test}} = 5720$ unique and unseen speaker mixtures from *LibriSpeech test-clean* set with $k_{\max} = 10$.

[4] Note that for the validation and testing, excerpts are fixed.

Several well-established input representations were evaluated in [76] such as (linear or logarithmically scaled) short-time Fourier transform (STFT), Mel filter bank outputs (MEL), Mel Frequency Cepstral Coefficients (MFCC) representations, typically chosen for speech applications.

Even though MFCCs are used in related tasks and are included in our baseline evaluations, they are known to perform poorly when used in CNNs [70]. This is why we decided to not to use the MFCCs as an input for the proposed architectures. The remaining input representations are identical to those listed in [76]:

1) STFT: magnitude of the Short-time Fourier transform computed using Hann-windows. A frame length of 25 ms has been used. The resulting input is $X \in \mathbb{R}^{500\times 201}$.
2) LOGSTFT: logarithmically scaled magnitudes from STFT representation using $\log(1+STFT)$. The resulting input is $X \in \mathbb{R}^{500\times 201}$.
3) MEL: compute mapping from the STFT output directly onto Mel basis using 40 triangular filters. The resulting input is $X \in \mathbb{R}^{500\times 40}$.

Before feature transformation, all input files were re-sampled to 16 kHz sampling rate. All features are computed using a hop size of 10 ms.

5.1 Metric

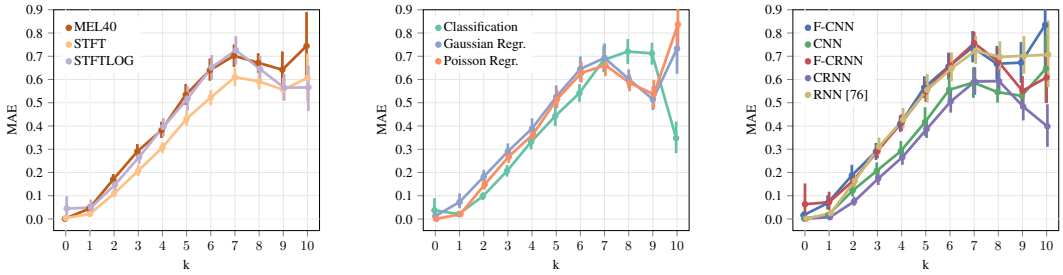
Whereas the intermediate output y is treated as either a classification or a regression problem (see Section 2) we evaluate the final output k as a discrete regression problem. We, therefore, employ the mean absolute error (MAE) which is also commonly used for other count related tasks (c.f. [61,86]). Since the MAE depends on the true count k, we also present the MAE per class as:

$$\mathrm{MAE}(k) = \frac{1}{T_{\text{test}}} \sum_{t=1}^{T_{\text{test}}} \left| \hat{k} - k \right|. \tag{11}$$

which is then averaged across the classes, i.e.,

$$\mathrm{MAE} = \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}} \mathrm{MAE}(k). \tag{12}$$

5.2 Model Comparison



(a) by feature representations. (b) by output distribution. (c) by architecture.

Figure 3: Average mean absolute error (MAE) on mixtures of speakers with equal power as described in SECTION 5.2 per ground truth count $k = [0 \dots 10]$. Error bars show the 95% confidence intervals. Results in (a) are averaged over factors shown in (b) and (c) and similarly for (b) and (c).

To find the best parameters we performed training and evaluation for different input representations and output distributions (c.f. [76]) as well as all proposed architectures resulting in 135 models. On average each model was trained for 25 epochs before early stopping was engaged. We present the results filtered by the three factors (Architecture, Input and Output) in Fig. 3. One can see that the overall trend of the count error in MAE is similar regardless of the parametrization: all models are able to reliably distinguish between $k = 0$ and $k = 1$, followed by a nearly linear

Table 3: Mixed Effects Linear Model for $k = \{1, 2 \dots 7\}$. Model: $MAE \sim architecture + feature + objective + (1|k)$

Factor	Coef.	Std.Err.	z	$P > \|z\|$
Intercept	0.305	0.091	3.360	0.001
architecture = CRNN	-0.028	0.011	-2.419	0.016
architecture = F-CNN	0.102	0.011	8.976	0.000
architecture = F-CRNN	0.102	0.011	8.947	0.000
architecture = RNN	0.094	0.011	8.240	0.000
feature = STFT	-0.079	0.009	-8.946	0.000
feature = STFTLOG	-0.001	0.009	-0.117	0.907
objective = P-Regression	0.040	0.009	4.555	0.000
objective = G-Regression	0.067	0.009	7.651	0.000
Random Effect k	0.057	0.297		

increase in MAE between $k = \{1, 2 \dots 7\}$. For $k > 7$ it can be seen that the classification type models have learned the maximum of k across the dataset, hence the prediction error decreases when k reaches its maximum. This is because classification based models intrinsically have access to the maximum number of sources determined by the output vector dimensionality. Furthermore, one can see that all three factors have only little effect on the overall performance of the model, which is especially the case for small k. As indicated by Fig. 3a, choosing linear STFT as input representation generally results in a better performance compared to MEL and even LOGSTFT. Concerning the output distribution, a similar observation can be made about classification which outperforms Poisson regression and Gaussian regression, as indicated by Fig. 3b. In Fig. 3c the performance of our proposed architectures are compared: while CNN and CRNN are close, both of them perform better than full frequency band F-CNN and F-CRNN models as well as the recurrent based architecture, proposed in [76]. However, it is interesting that, despite its simplicity, the F-CNN and F-CRNN, perform similarly to the Bi-LSTM architecture.

The results are supported by a statistical evaluation based on mixed effect linear model (see Table 3) where k is modeled as a random effect (for further details we refer to [49]). For a fair comparison (i.e. reducing the bias towards classification type network) of all models we only evaluate results for $k = \{1, 2 \dots 7\}$; however, all networks were trained on $k = \{0, \dots, 10\}$. These results indicate that CRNN performs statistically significantly better than the CNN. Concerning the input representation, we can report that using STFT representation outperforms the log-scaled STFT as well as the MEL representation. Interestingly, we did not find any significant differences between MEL and STFTLOG in MAE performance. With respect to the output distributions, we can report that Classification outperforms the other two distributions while Poisson regression performs better than Gaussian regression which confirms the findings made in [76] based on the RNN model. Therefore, we select the CRNN classification model with STFT features for subsequent experiments.

Figure 4 gives an indication of the efficiency of each model and the trade-off between performance and complexity in terms of parameters and floating point multiplications. It can be seen that the CRNN is not only the one that performs best but also has significantly fewer parameters than the CNN model. In contrast, the F-CRNN model does only have a fraction of the number of parameters of the other models, which makes it the most suitable model for mobile applications.

6 Evaluation Results

In this section, we perform several experiments on the proposed CRNN model that has been selected in the previous section. We assess the performance of this model by showing the results of three experiments that augment the test data by choosing a different dataset, varying amplitude gain levels and introduce reverberation. These results also include several baseline methods. Furthermore, we present the effect of training sample duration and compare the results from the DNN to human performance gathered in a listening experiment.

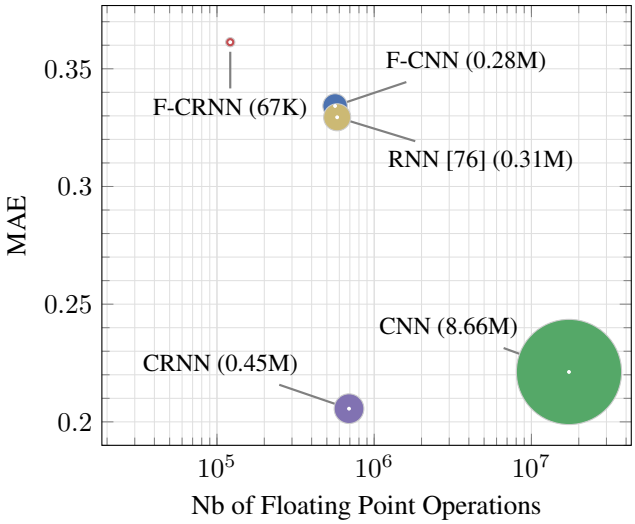


Figure 4: Complexity in number of floating point multiplications and number of weight parameters (in brackets) over performance in MAE of our five proposed models.

6.1 Baselines

In order to make a meaningful comparison to the CRNN model we propose several baseline methods. Since we are dealing with a novel task description, related speaker count estimation techniques like those introduced in Section 1 can hardly be used as baselines. Specifically, [82] would not work on fully overlapped speech, [5] does not scale to the size of our dataset, since it requires to cross-correlate the full database against another. Finally, [65] proposes a feature but does not employ a fully automated system that can be used in a data-driven context. We, therefore, decided to propose our own baseline methods.

VQ This method uses a feature proposed by Sayoud [65] based on 7th MEL filter coefficient (MFCC_7) which was shown to encode sufficiently important speaker-related information. The temporal dimension of X is squashed down by subtracting the mean and standard deviation as $X = \overline{\text{MFCC}_7} - STD(\text{MFCC}_7) \in \mathbb{R}^1$. In [65] the mapping from $X \Rightarrow k$ is done by manually thresholding X. To translate this into a data-driven approach, we employed a vector quantizer (using k-means) to get an optimal mapping with respect to the sum of squares criterion. Further, as preprocessing, we added the same normalization as for our proposed CRNN which in turn decreases the performance of the method significantly as it is highly gain dependent.

SVM, SVR We found that the information encoded in the 7th MFCC coefficient as used in the **VQ** baseline, may not suffice to explain the high variability in our dataset. This is especially important for larger speaker counts. We therefore extended VQ by including all 20 MFCCs but using the same temporal dimensionality reduction, resulting in $X = \overline{MFCC} - STD(MFCC) \in \mathbb{R}^{20}$. To deal with significantly increased dimensionality of X, we used a support vector machine (SVM) with a radial basis function (RBF) kernel. Similarly to our proposed DNN based methods, we treat the output as either a classification problem or a regression problem through the use of support vector regression (SVR).

6.2 Results on Gain Variations

In our parameter optimization in Section 5 we evaluated mixtures with speakers having equal power. In a more realistic scenario, speakers often differ in volume between utterances. We simulate this by introducing gain factors between 0.5 and 2.0, randomly applied to the sources, hence resulting in a deviation of 6 dB compared to the reference where all speakers are mixed to have equal power. We applied this variation only to the test data to evaluate how models generalize to this updated condition. The results of this experiment are presented in Table 4. **MEAN** corresponds to the case

Table 4: Averaged MAE results of different methods on several datasets for $k = [0 \dots 10]$ with equal power and random gains (up to ± 6 dB)) as well as reverberation. Bold face indicates the best-performing method.

Trained on	*LIBRI*							*LIBRI-Reverb*
Test Set	LIBRI			THCS10		TIMIT		LIBRI-Reverb
Variation	–	± 6 dB	Reverb	–	± 6 dB	–	± 6 dB	Reverb
CRNN	$\mathbf{0.27 \pm 0.22}$	$\mathbf{0.43 \pm 0.39}$	1.63 ± 0.22	$\mathbf{0.36 \pm 0.25}$	$\mathbf{0.50 \pm 0.46}$	$\mathbf{0.31 \pm 0.33}$	$\mathbf{0.52 \pm 0.52}$	$\mathbf{0.48 \pm 0.22}$
RNN [76]	0.38 ± 0.28	0.57 ± 0.49	1.41 ± 0.87	0.58 ± 0.50	0.76 ± 0.72	0.48 ± 0.41	0.72 ± 0.65	0.59 ± 0.43
SVR	0.58 ± 0.27	0.61 ± 0.31	$\mathbf{0.76 \pm 0.35}$	0.69 ± 0.28	0.73 ± 0.32	0.70 ± 0.45	0.62 ± 0.36	0.71 ± 0.35
SVC	0.63 ± 0.39	0.66 ± 0.37	0.85 ± 0.51	0.77 ± 0.37	0.77 ± 0.36	0.89 ± 0.75	0.76 ± 0.61	0.78 ± 0.45
VQ [65]	2.41 ± 1.08	2.41 ± 1.06	2.41 ± 1.08	2.98 ± 1.62	2.98 ± 1.60	2.13 ± 1.06	2.15 ± 1.07	2.41 ± 1.13
MEAN	2.73 ± 1.63	2.73 ± 1.63	2.73 ± 1.63	2.73 ± 1.64	2.73 ± 1.63	2.73 ± 1.63	2.73 ± 1.63	2.73 ± 1.63

when $k = 5$ is predicted for all test samples. Our results indicate that augmenting the mixture gains does have an impact on performance, for both, our proposed CRNN model as well as the baseline methods. For example, for the CRNN model the performance drops by 60% from 0.27 MAE to 0.43 MAE on the *LIBRI Speech* test set, which is still about 40% better than the second best-performing method *SVR* which drops from 0.58 MAE to 0.61 MAE.

6.3 Results on Different Datasets

We also present results on two additional datasets. Again, we only changed the test data; all networks were trained on *LIBRI Speech*. Compared to *LIBRI Speech*, the *TIMIT* database has an overall lower recording quality. This is reflected by our results where the performance in MAE drops only slightly between these two datasets. Interestingly, even when we look at the results of the Mandarin language *THCS10* dataset, performance drops only slightly. More precisely, for our proposed CRNN model, test performance on *THCS10* is even better than on its own *LIBRI* dataset with gain variations. These results suggest that the trained model is speaker and language independent.

6.4 Effect of Reverberant Signals

Different acoustical conditions such as increased reverberation time were shown [55] to have a large effect in speaker counting. To analyze this effect, different acoustic conditions were simulated by generating the room impulse responses using the image method [2, 29]. For this experiment we set up an acoustical room with dimension (3.5 m $\times$ 4.5 m $\times$ 2.5 m) The microphone was positioned at (1m, 1m, 1m). For the mentioned room, 350 different reverberation times were selected uniformly sampled between 0.1 and 0.5 seconds. For each of these reverberation times, we generated unique room impulse responses that correspond to individual source positions which have minimum distance 0.1 m to the walls and are otherwise positioned randomly on the (X, Y, 1m) plane. Each speaker's signal was convolved with a randomly selected room impulse response before mixing. Results, again, are shown in Table 4. For the first time, we can see that the CRNN model significantly drops in performance from 0.27 MAE to 1.64 MAE, whereas the SVR and SVM baselines are only affected slightly. This is expected as these baselines are using a temporal aggregation of all frames, whereas the CRNN is based on smaller (3×3) convolutional filter operations that are able to capture the room acoustics as well. If we assume that our trained deep learning model is fully speaker independent, a mixture of two utterances from the same speaker would get the same count estimate as two different speakers. Hence, reverberation tends to result in overestimation and we observed this even for $k = 1$ where it, in turn, resulted in an increase in MAE.

To further investigate whether the overestimation can be reduced via training with reverberant samples, we created a separate set of room impulse responses for the training dataset with different room dimensions so that the model cannot learn the acoustical conditions from the training dataset. From the results shown in the last column of Table 4 we can see that the retrained CRNN is able to outperform the baselines again. Therefore, when retrained with reverberant samples, the proposed model is able to better discriminate between a reverberant component of the same speaker and contributions from different speakers. For robustness against different acoustic conditions, it is essential to include reverberant samples in the training dataset.

6.5 Effect of Duration and Overlap Detection Error

In our last experiment we want to address the influence of the input duration length D. In a real-world application this parameter would be chosen as small as a possible, because a longer input duration adds both algorithmic and computational delay to a real-time system. In a small experiment, we took the proposed CRNN and retrained it using a different number of input frames ranging from 100 to 900 frames (corresponding to one to nine seconds of audio). For each input duration, we trained the CRNN with three different initial seeds. Results are shown in Fig. 5. It can be seen that five second duration is a good trade-off between performance and delay. If latency is critical, keeping D above 2 seconds is recommended for good results. For segments as short as 1 second the MAE of around 0.6 is almost twice as high as for segments of 5 seconds duration. However, if instead of the count estimation MAE we compute the accuracy to detect overlap $k > 1$ vs. non-overlap $k \in 0, 1$, we still achieve 98.7% accuracy (precision: 99.7%, recall: 98.7%). This shows that our system can be effectively used to address overlap detection.

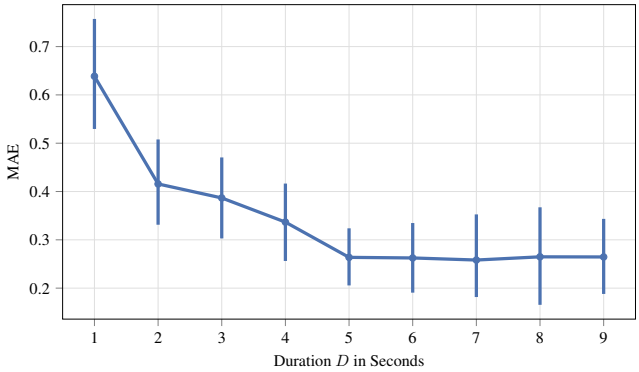


Figure 5: Evaluation of trained CRNN networks over different input duration length D. Error bars show 95% confidence intervals.

6.6 Listening Experiment

To compare the results of our trained CRNN on our synthesized dataset to human performance, we chose to reproduce the experiments made in [41, 42]. Kawashima et al. found in extensive experiments using Japanese speech samples, that participants were able to correctly estimate up to three simultaneously active speakers without using any spatial cues. We conducted our own study using the simulated data from the *LIBRI Speech* (power normalized) set mentioned earlier in Section 4.1. We therefore randomly selected 10 samples for each $k \in [0, \ldots, 10]$, resulting in 100 mixtures of 5 seconds duration each. The experiment was done using *between-group design*, where one group (blind experiment) did not get any prior information about the maximum number of speakers in the test set (similar to [42]). However, the maximum number of speakers was revealed to the other group (informed experiment), which is more related to our data-driven, classification based CRNN. Further, none of the participants received any feedback about the error made during the trials. Similarly to [42], lab-based experiments were conducted with ten participants for each group ($n = 20$) using a custom designed web-based software.[5] In all previous experiments, we used the mean absolute error metric which does not reveal over and underestimation errors. We therefore decided to report the average response for each group of k. The results of our lab-based experiments are shown in Fig. 6. The results for up to three speakers indicate that humans perform similarly (or better in terms of variance) compared to our proposed CRNN model. Results of the blind experiment show that underestimation becomes apparent for $k > 3$. As a reference, we also included the average results from [42] (Experiment 1, 5 seconds durations) which shows similar results compared to our blind experiment. For larger speaker counts, the gap between humans and algorithm is almost three speakers on average. Interestingly, the results of the informed experiment reveal that this gap closes down to an average difference of one speaker. Finally, we can report that the machine model reached superhuman performance. Unlike humans, the CRNN is subject to over-estimations for $4 < k \leq 9$. However, with extensive training, humans might be able to perform

[5]The experiment is made available through the accompanying website.

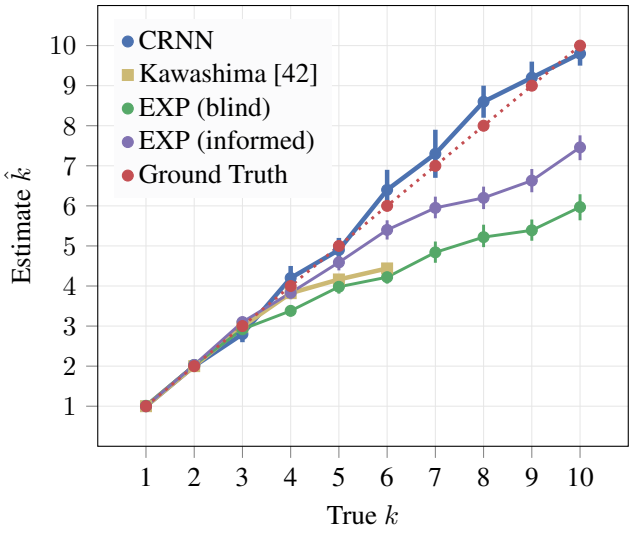


Figure 6: Average responses from humans (EXP and *Kawashima* [42]) compared to our proposed CRNN. Error bars show 95% confidence intervals.

on par. When we asked participants about the strategy they pursued, many reported that with more than three speakers it is not possible to identify (and count) the speakers but rather compare the *density* of the speech to that of 1-3 speakers. For higher speaker counts, participants reported that the integrated phoneme activity was a relevant cue, supporting our previously mentioned hypothesis.

7 Understanding CountNet

In this section, we focus on the problem of interpreting the strategy undergone by this system for successful counting.

7.1 Saliency Maps

We first conducted a visual analysis based on salience map representations [72]. In the deep learning context, saliency maps are visualizations that are able to show which specific input elements a neural network used for a particular prediction. This allows an object classifier to be used for object localization or in the case of audio spectrograms, which time-frequency bins are most relevant. The common idea is to compute the gradient of the model's prediction with respect to the input, holding the weights fixed. This determines which input elements need to be changed the least to affect the prediction the most.

In this work, we used guided backpropagation, first introduced in [74] and successfully deployed in [66] to compute a saliency map for singing voice detection. For a given input of a three-speaker mixture, we depicted the saliency map in Fig. 7. The saliency map indicates that our proposed model does not rely much on the overlapped parts but instead utilize many of the single speaker time-frequency bins as well as many high-frequency components such as plosives and fricative phonemes.

While the saliency map confirms that the network does exploit both low and high-frequency content from the input signal, it is not sufficient to conjecture about the strategy implemented in the network.

7.2 Ablation Analysis

To provide further insight, we propose another layer-wise analysis, that provides information concerning the behavior of the model at different successive layers. While we cannot show all filter outputs (e.g. 64, for the first layer), instead, for each filter, we compute its loss with respect to the input of the model using gradient update and sort the filters according to their loss behavior.

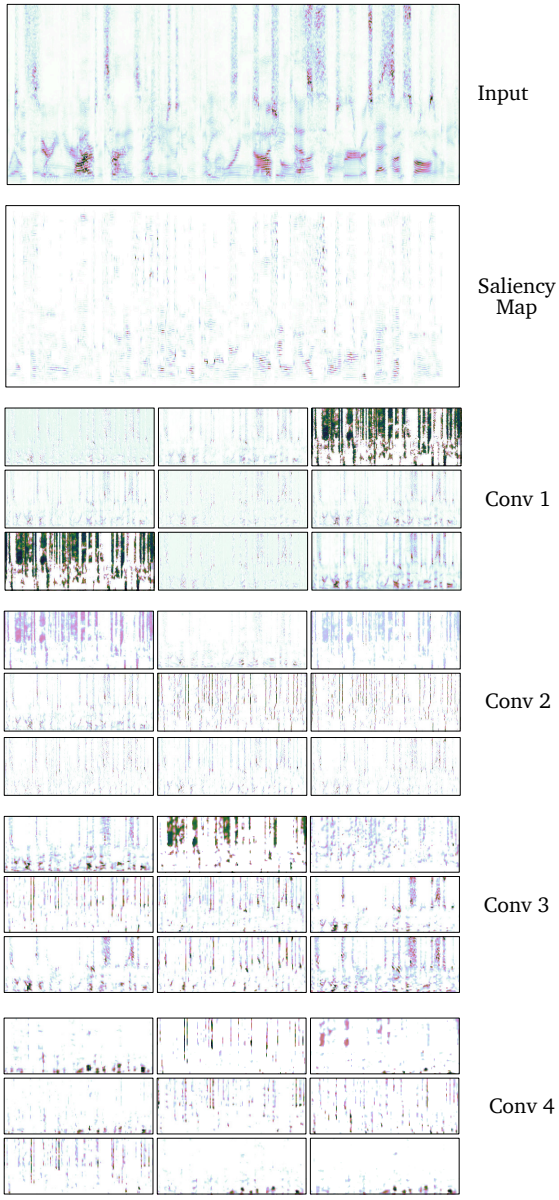


Figure 7: Illustration of intermediate outputs from the proposed CRNN for each convolutional layer for a given input with $k = 3$ speakers. Saliency map shows positive saliency of guided backpropagation [74]. For each convolutional layer the nine most relevant filters were selected based on their loss with respect to the input.

Figure 7 depicts the nine highest loss outputs per convolutional layer. We can observe that while the first layer shows only low-level variations of the input, already the second layer seems to be more abstract and emphasizes phoneme segmentations based on mid and high frequency content. While filter outputs of layer 3 and 4 also show more low-frequency content such as the harmonic signals, the overall visual impression is that the proposed CRNN focuses on the temporal segmentation of phonemes.

The conducted analysis suggests that the network is doing count estimation based on the detection of phonemes. To assess the validity of this interpretation, we directly verified the performance of the

Table 5: Results of a binary logit regression test for the dependent variable *correct response* over the independent variable *speaking rate*. The results are based on $n = 2000$ randomly drawn results of the CRNN model trained and evaluated on the TIMIT dataset.

	coef	**std err**	**z**	**P>|z|**
speaking rate	-1.2697	0.232	-5.477	0.000
intercept	4.3213	0.790	5.468	0.000

method as a function of the phoneme activity. In the following, we verify whether count estimates are affected by the pronunciation speed.

We assume that the CRNN model learned the aggregated phoneme or syllable activity of all speakers in a fixed, given excerpt. If that is the case, it would mean that the speaker count estimate would be affected if the speakers would speak slower or faster in relation to the fixed input window (speaking rate). We therefore want to see if very slow or very fast speakers significantly increase the error of our proposed CRNN model. In turn we define a null hypothesis that there is no association between the speaker count error probability and the value of the *speaking rate*.

To verify our hypothesis, we created another experiment based on the *TIMIT* dataset. It comes with phoneme and word level annotations, from which the speaking rate (defined as syllables per second) can be computed for each input sample [40]. To reduce the influence of the different acoustical environment in TIMIT compared to Libri Speech, we retrained the CRNN classification model on the *TIMIT* training dataset, using the same parameters as described in Section 4. At test time we randomly generated 5 seconds excerpts of $k = 6$ from the TIMIT test subset and predicted the error $E(k) = \hat{k} - k$ for each CRNN output. We grouped the estimates into three classes: $E(k) = 0$ (correct response), $E(k) > 0$ (overestimation), $E(k) < 0$ (underestimation). For $k = 6$ we ended up with two groups of results because overestimation did not take place. From the remaining two groups *underestimation* and *correct* responses we randomly selected 1000 samples each, resulting in an total sample size of $n = 2000$. For these samples we computed an average speaking rate of 3.40 syllables per second and a standard deviation of 0.2.

We chose a Generalized Linear Model (GLM) for the statistical test, as described in [38]. This allows us model the results with a binary logit regression model that turns the mean of E into a binomial distributed probability modeled by log linear values: $\text{logit}(E) \sim \text{Intercept} + \beta \cdot \text{Speaking Rate}$. The results of our test are shown in Table 5 and indicate the speaking rate has statistically significant influence on the error $p < 0.05, df = 1, \text{Pseudo } R^2 = 0.0111$. To better understand the effect of our predictor, we computed an odds ratio $\exp(\text{speaking rate}) = 0.28$.

This indicates that a decrease in speaking rate of 1 syllable per second will increase the likeliness of an underestimation error by 28 percent. Even though this is considered as a small effect size, it gives an interesting hint for the strategy taken of our proposed model and also suggests that for improved robustness, training would benefit from a large variety of speaking rates. Furthermore, it still remains unclear the model would suffer from languages with a speaking rate which is naturally higher or lower than English or Chinese (see [53]).

8 Conclusion

We introduced the task of estimating the maximum number of concurrent speakers in a simulated "cocktail-party" environment using a data-driven approach, discussing how to frame this task in a deep learning context. Building upon earlier work, we investigated what method is best to output integer source count estimates and also defined suitable cost functions for optimization. In a comprehensive study, we performed experiments to evaluate different network architectures. Furthermore, we investigated and evaluated other important parameters such as input representations or the input duration. Our final proposed model uses a convolutional recurrent (CRNN) architecture, based on classification at the network's output. Compared to several baselines, our proposed model has a significantly lower error rate; it achieves error rates of less than 0.3 speakers in mean absolute error for classifying zero to ten speakers—a decrease of 28.95% compared to [76]. In further simulations, we revealed that our model is robust to unseen languages (such as Chinese), as well

as varying acoustical conditions (except for reverberation, where the error increased significantly). However, including reverberated samples in the training reduces the error. Additionally, we conducted a perceptual experiment showing that these results clearly outperform humans. We hope our research stimulates future research on data-driven count estimation, a task that currently lacks real-world datasets. Lastly, in an ablation study, we found that the CRNN uses a strategy to segment phonemes/syllables to estimate the count. Hence, we hypothesize that a speaker count estimate is influenced by the average speaking rates of certain languages. Finally, to underpin this hypothesis, we showed that the speaking rate has a significant effect on the error of our model.

Acknowledgments

The authors gratefully acknowledge the compute resources and support provided by the Erlangen Regional Computing Center (RRZE).

Many thanks to Antoine Liutkus for his constructive criticism of the manuscript.

References

[1] Webrtc vad v2.0.10. `https://github.com/wiseman/py-webrtcvad`.

[2] J. B. Allen and D. A. Berkley. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America*, 65(4):943–950, 1979.

[3] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos, et al. Deep speech 2: End-to-end speech recognition in English and Mandarin. In *Proc. Intl. Conference on Machine Learning*, pages 173–182, 2016.

[4] V. Andrei, H. Cucuand, and C. Burileanu. Detecting overlapped speech on short timeframes using deep learning. In *Proc. Interspeech Conf.*, 2017.

[5] V. Andrei, H. Cucuand, A. Buzo, and C. Burileanu. Counting competing speakers in a time frame - human versus computer. In *Proc. Interspeech Conf.*, 2015.

[6] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals. Speaker diarization: A review of recent research. *IEEE Trans. Audio, Speech, Lang. Process.*, 20(2):356–370, 2012.

[7] T. Arai. Estimating number of speakers by the modulation characteristics of speech. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, volume 2, pages II–197, 2003.

[8] S. Araki, T. Nakatani, H. Sawada, and S. Makino. Stereo source separation and source counting with map estimation with dirichlet prior considering spatial aliasing problem. In *Proc. Intl. Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, pages 742–750. Springer, 2009.

[9] S. Arberet, R. Gribonval, and F. Bimbot. A robust method to count and locate audio sources in a multichannel underdetermined mixture. *IEEE Trans. Signal Process.*, 58(1):121–133, 2010.

[10] C. Arteta, V. Lempitsky, and A. Zisserman. Counting in the wild. In *European Conference on Computer Vision*, pages 483–498. Springer, 2016.

[11] J. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, 1985.

[12] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *Advances in neural information processing systems*, pages 2546–2554, 2011.

[13] K. Boakye, O. Vinyals, and G. Friedland. Two's a crowd: Improving speaker diarization by automatically identifying and excluding overlapped speech. In *Proc. Interspeech Conf.*, 2008.

[14] L. Boominathan, S. S. Kruthiventi, and R. V. Babu. Crowdnet: A deep convolutional network for dense crowd counting. In *Proc. ACM Intl. Conference on Multimedia (ACMMM)*, pages 640–644. ACM, 2016.

[15] A. S. Bregman. *Auditory scene analysis: The perceptual organization of sound*. MIT press, 1994.

[16] A. B. Chan and N. Vasconcelos. Bayesian poisson regression for crowd counting. In *Proc. IEEE Intl. Conference on Computer Vision (ICCV)*, pages 545–551, 2009.

[17] P. Chattopadhyay, R. Vedantam, R. R. Selvaraju, D. Batra, and D. Parikh. Counting everyday objects in everyday scenes. In *Proc. Intl. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[18] K. Choi, G. Fazekas, M. Sandler, and K. Cho. Convolutional recurrent neural networks for music classification. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2392–2396, March 2017.

[19] K. P. Choi. On the medians of gamma distributions and an equation of ramanujan. *Proceedings of the American Mathematical Society*, 121(1):245–251, 1994.

[20] F. Chollet et al. Keras v1.2.2. `https://github.com/fchollet/keras/tree/1.2.2`, 2015.

[21] S. Dieleman and B. Schrauwen. End-to-end learning for music audio. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6964–6968, May 2014.

[22] L. Drude, A. Chinaev, D. H. T. Vu, and R. Haeb-Umbach. Source counting in speech mixtures using a variational EM approach for complex watson mixture models. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6834–6838, 2014.

[23] N. Fallah, H. Gu, K. Mohammad, S. A. Seyyedsalehi, K. Nourijelyani, and M. R. Eshraghian. Nonlinear poisson regression using neural networks: a simulation study. *Neural Computing and Applications*, 18(8):939, 2009.

[24] D. Garcia-Romero, D. Snyder, G. Sell, D. Povey, and A. McCree. Speaker diarization using deep neural network embeddings. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4930–4934, Mar. 2017.

[25] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren. DARPA TIMIT acoustic phonetic continuous speech corpus CDROM, 1993.

[26] J. T. Geiger, F. Eyben, B. W. Schuller, and G. Rigoll. Detecting overlapping speech with long short-term memory recurrent neural networks. In *Proc. Interspeech Conf.*, pages 1668–1672, 2013.

[27] E. M. Grais and M. D. Plumbley. Single channel audio source separation using convolutional denoising autoencoders. In *Proc. GlobalSIP*, pages 1265–1269, Nov. 2017.

[28] G. Gravier, G. Adda, N. Paulson, M. Carr'e, A. Giraudel, and O. Galibert. The ETAPE Corpus for the Evaluation of Speech-based TV Content Processing in the French Language. In *LREC - Eighth international conference on Language Resources and Evaluation*, Turkey, 2012.

[29] E. A. P. Habets. Room impulse response (RIR) generator. `https://github.com/ehabets/RIR-Generator`, 2016.

[30] G. Hagerer, V. Pandit, F. Eyben, and B. Schuller. Enhancing lstm rnn-based speech overlap detection by artificially mixed data. In *Proc. Audio Eng. Soc. Conference on Semantic Audio*, June 2017.

[31] J. Hershey, Z. Chen, J. Le Roux, and S. Watanabe. Deep clustering: Discriminative embeddings for segmentation and separation. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 31–35, Mar. 2016.

[32] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[33] S. Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 6(2):107–116, Apr. 1998.

[34] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997.

[35] M. Hrúz and M. Kunešová. Convolutional neural network in the task of speaker change detection. In *International Conference on Speech and Computer*, pages 191–198. Springer, 2016.

[36] M. Huijbregts, D. A. van Leeuwen, and F. Jong. Speech overlap detection in a two-pass speaker diarization system. In *Proc. Interspeech Conf.*, Brighton, 2009.

[37] D. Huron. Voice denumerability in polyphonic music of homogeneous timbres. *Music Perception: An Interdisciplinary Journal*, 6(4):361–382, 1989.

[38] T. F. Jaeger. Categorical data analysis: Away from ANOVAs (transformation or not) and towards logit mixed models. *Journal of memory and language*, 59(4):434–446, 2008.

[39] W. S. Jevons. The power of numerical discrimination. *Nature*, 3(67):281–282, 1871.

[40] Y. Jiao, M. Tu, V. Berisha, and J. Liss. Online speaking rate estimation using recurrent neural networks. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5245–5249, March 2016.

[41] M. Kashino and T. Hirahara. One, two, many – judging the number of concurrent talkers. *J. Acoust. Soc. Am.*, 99(4):2596–2603, 1996.

[42] T. Kawashima and T. Sato. Perceptual limits in a simulated cocktail party. *Attention, Perception and Psychophysics*, 77(6):2108–2120, 2015.

[43] A. Khan, S. Gould, and M. Salzmann. Deep convolutional neural networks for human embryonic cell counting. In *European Conference on Computer Vision*, pages 339–348. Springer, 2016.

[44] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proc. ICLR*, 2014.

[45] A. Lefevre, F. Bach, and C. Févotte. Itakura-saito nonnegative matrix factorization with group sparsity. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 21–24, 2011.

[46] S. Leglaive, R. Hennequin, and R. Badeau. Singing voice detection with deep recurrent neural networks. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 121–125, Apr. 2015.

[47] B. Loesch and B. Yang. Source number estimation and clustering for underdetermined blind source separation. In *Proc. Intl. Workshop Acoust. Echo Noise Control (IWAENC)*, 2008.

[48] M. Marsden, K. McGuiness, S. Little, and N. E. O'Connor. Fully convolutional crowd counting on highly congested scenes. In *12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, 2017.

[49] C. E. Mcculloch and J. M. Neuhaus. *Generalized Linear Mixed Models*. 2006.

[50] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen. DCASE 2017 challenge setup: Tasks, datasets and baseline system. In *DCASE 2017-Workshop on Detection and Classification of Acoustic Scenes and Events*, 2017.

[51] A. Mesaros, T. Heittola, and T. Virtanen. TUT database for acoustic scene classification and sound event detection. In *Proc. European Signal Processing Conf. (EUSIPCO)*, Budapest, Hungary, 2016.

[52] S. Mirzaei, Y. Norouzi, et al. Blind audio source counting and separation of anechoic mixtures using the multichannel complex NMF framework. *Signal Processing*, 115:27–37, 2015.

[53] H. Osser and F. Peng. A cross cultural study of speech rate. *Language and Speech*, 7(2):120–125, 1964.

[54] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: An ASR corpus based on public domain audio books. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015.

[55] S. Pasha, J. Donley, and C. Ritz. Blind speaker counting in highly reverberant environments by clustering coherence features. In *Asia-Pacific Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC). IEEE*, Dec. 2017.

[56] D. Pavlidi, A. Griffin, M. Puigt, and A. Mouchtaris. Source counting in real-time sound source localization using a circular microphone array. In *IEEE Signal Processing Workshop on Sensor Array and Multichannel (SAM)*, pages 521–524, 2012.

[57] C. Pierre and C. Jutten. *Handbook of Blind Source Separation*. Academic Press, 2010.

[58] J. Pons, T. Lidy, and X. Serra. Experimenting with musically motivated convolutional neural networks. In *Intl. Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6, 2016.

[59] J. Pons, O. Slizovskaia, R. Gong, E. Gómez, and X. Serra. Timbre analysis of music audio signals with convolutional neural networks. *Proc. European Signal Processing Conf. (EUSIPCO)*, 2017.

[60] V. S. Ramaiah and R. R. Rao. Speaker diarization system using HXLPS and deep neural network. *Alexandria Engineering Journal*, 2017.

[61] S. H. Rezatofighi, V. K. BG, A. Milan, E. Abbasnejad, A. Dick, and I. Reid. DeepSetNet: Predicting sets with deep neural networks. In *Proc. IEEE Intl. Conference on Computer Vision (ICCV)*, 2017.

[62] M. Rouvier, P.-M. Bousquet, and B. Favre. Speaker diarization through speaker embeddings. In *Proc. European Signal Processing Conf. (EUSIPCO)*, pages 2082–2086, 2015.

[63] M. Rouvier, G. Dupuy, P. Gay, E. Khoury, T. Merlin, and S. Meignier. An open-source state-of-the-art toolbox for broadcast news diarization. In *Proc. Interspeech Conf.*, 2013.

[64] T. Sainath, O. Vinyals, A. Senior, and H. Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4580–4584, 2015.

[65] H. Sayoud and S. Ouamour. Proposal of a new confidence parameter estimating the number of speakers-an experimental investigation. *Journal of Information Hiding and Multimedia Signal Processing*, 1(2):101–109, 2010.

[66] J. Schlüter. Learning to pinpoint singing voice from weakly labeled examples. In *Proc. Intl. Society for Music Information Retrieval Conference (ISMIR)*, pages 44–50, 2016.

[67] J. Schlüter and T. Grill. Exploring data augmentation for improved singing voice detection with neural networks. In *Proc. Intl. Society for Music Information Retrieval Conference (ISMIR)*, pages 121–126, 2015.

[68] M. Schoeffler, F.-R. Stöter, H. Bayerlein, B. Edler, and J. Herre. An experiment about estimating the number of instruments in polyphonic music: a comparison between internet and laboratory results. In *Proc. Intl. Society for Music Information Retrieval Conference (ISMIR)*, pages 389–394, 2013.

[69] S. Seguí, O. Pujol, and J. Vitria. Learning to count with deep object features. In *Proc. Intl. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 90–96, 2015.

[70] M. L. Seltzer, D. Yu, and Y. Wang. An investigation of deep neural networks for noise robust speech recognition. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7398–7402, May 2013.

[71] N. Shokouhi and J. H. L. Hansen. Teager–kaiser energy operators for overlapped speech detection. *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, 25(5):1035–1047, May 2017.

[72] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

[73] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. ICLR*, 2015.

[74] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller. Striving for simplicity: The all convolutional net. *CoRR*, abs/1412.6806, 2014.

[75] F.-R. Stöter, M. Schoeffler, B. Edler, and J. Herre. Human ability of counting the number of instruments in polyphonic music. In *Proceedings of Meetings on Acoustics*, volume 19, 2013.

[76] F.-R. Stöter, S. Chakrabarty, B. Edler, and E. A. P. Habets. Classification vs. regression in supervised learning for single channel speaker count estimation. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[77] G. t. Hoopen and J. Vos. Effect on numerosity judgment of grouping of tones by auditory channels. *Attention, Perception, & Psychophysics*, 26(5):374–380, 1979.

[78] S. Uhlich, F. Giron, and Y. Mitsufuji. Deep neural network based instrument extraction from music. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2135–2139, April 2015.

[79] O. Walter, L. Drude, and R. Haeb-Umbach. Source counting in speech mixtures by nonparametric bayesian estimation of an infinite Gaussian mixture model. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 459–463, 2015.

[80] C. Wang, H. Zhang, L. Yang, S. Liu, and X. Cao. Deep people counting in extremely dense crowds. In *Proc. ACM Intl. Conference on Multimedia (ACMMM)*, pages 1299–1302, 2015.

[81] D. Wang, X. Zhang, and Z. Zhang. THCHS-30 : A free chinese speech corpus, 2015.

[82] C. Xu, S. Li, G. Liu, Y. Zhang, E. Miluzzo, Y.-F. Chen, J. Li, and B. Firner. Crowd++: Unsupervised speaker count with smartphones. In *Proc. of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 43–52. ACM, 2013.

[83] S. H. Yella, A. Stolcke, and M. Slaney. Artificial neural network features for speaker diarization. In *IEEE Workshop on Spoken Language Technology (SLT)*, pages 402–406, 2014.

[84] R. Yin, H. Bredin, and C. Barras. Speaker change detection in broadcast TV using bidirectional long short-term memory networks. In *Proc. Interspeech Conf.* ISCA, 2017.

[85] D. Yu, M. Kolbæk, Z.-H. Tan, and J. Jensen. Permutation invariant training of deep models for speaker-independent multi-talker speech separation. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[86] C. Zhang, H. Li, X. Wang, and X. Yang. Cross-scene crowd counting via deep convolutional neural networks. In *Proc. Intl. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 833–841, 2015.

[87] J. Zhang, S. Ma, M. Sameki, S. Sclaroff, M. Betke, Z. Lin, X. Shen, B. Price, and R. Mech. Salient object subitizing. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CCVPR)*, pages 4045–4054, 2015.

[88] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen. Convolutional recurrent neural networks for polyphonic sound event detection. *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, 25(6):1291–1303, June 2017.