Article

Learning Numerosity Representations with Transformers

Tommaso Boccato [1], Alberto Testolin [1,2,*] and Marco Zorzi [1,3,*]

[1] Department of General Psychology, University of Padova, Italy
[2] Department of Information Engineering, University of Padova, Italy
[3] IRCCS San Camillo Hospital, Venice-Lido, Italy
* Correspondence: alberto.testolin@unipd.it; marco.zorzi@unipd.it

Abstract: One of the most rapidly advancing areas of deep learning research aims at creating models that learn to disentangle the latent factors of variation from a data distribution. However, modeling joint probability mass functions is usually prohibitive, which motivates the use of conditional models assuming that some information is given as input. In the domain of numerical cognition, deep learning architectures have successfully demonstrated that approximate numerosity representations can emerge in multi-layer networks that build latent representations of a set of images with a varying number of items. However, existing models have focused on tasks requiring to conditionally estimate numerosity information from a *given image*. Here we focus on a set of much more challenging tasks, which require to conditionally generate synthetic images containing a *given number* of items. We show that attention-based architectures operating at the pixel level can learn to produce well-formed images approximately containing a specific number of items, even when the target numerosity was not present in the training distribution.

Keywords: Deep neural networks; Disentangled representations; Attention mechanisms; Generative models; Density estimation; Out-of-distribution generalization; Numerical cognition; Visual perception; Cognitive modeling

1. Introduction

In recent years, there has been a growing interest in the challenging problem of unsupervised representation learning [1]. Compared to the first wave of supervised deep learning success [2], unsupervised learning has great potential to further improve the capability of artificial intelligence systems, since it would allow building high-level, flexible representations without the need of explicit human supervision. Unsupervised deep learning models are also plausible from a cognitive [3] and biological [4] perspective, because they suggest how the brain could extract multiple levels of representations from the sensory signal by learning a hierarchical generative model of the environment [5–8].

Early approaches based on deep belief networks [9] already established that unsupervised representation learning leads to the discovery of high-level visual features, such as object parts [10] or written shapes [11,12]. However, the full potential of deep generative models was revealed by the introduction of variational autoencoders (VAE) [13] and generative adversarial networks (GAN) [14], which can discover and factorize extremely abstract attributes from the data [15,16]. These architectures can be further extended to promote the emergence of even more disentangled representations, such as in beta-VAE [17] and InfoGAN [18], or can exploit attention mechanisms to produce meaningful decompositions of complex visual scenes [19].

An interesting case study to investigate the representational capability of deep learning models is that of *numerosity perception*, which consists of rapidly estimating the number of objects in a visual scene without resorting to sequential counting procedures [20]. Compared to other high-level visual features, numerosity information is particularly challenging to extract because it refers to a global property of the visual scene, which co-varies with many other non-numerical visual features such as cumulative area, density and item size [21]. The emergence of numerosity representations has been successfully simulated using deep belief networks [22–24], which can approximately estimate the number of items in a given image (matching human-level performance) and partially disentangle it from non-numerical magnitudes [25]. However, learning fully disentangled representations of numerosity seems to be still out of reach even for state-of-the-art generative models, such as the InfoGAN [26].

In this paper we investigate whether the deployment of *self-attention mechanisms* allows to more precisely encode numerosity information as a disentangled factor of variation. Attention mechanisms [27] were first introduced in the context of machine translation to overcome the limitations of sequence-to-sequence architectures [28], which aimed at compressing the information contained in temporal sequences into fixed-length latent vectors. Shortly after, a novel architecture based solely on attention called *Transformer* [29] achieved new heights by completely dropping recurrence and convolutions. In analogy with the dynamics of associative memories [30], the power of this approach lies in the possibility of using a global attention mechanism to precisely and adaptively weight the contribution of each input element during processing. Transformers are starting to be applied also outside the language domain, with notable success in challenging computer vision tasks [31–33].

These promising results motivated our present work. In particular, we demonstrate that attention mechanisms can be successfully exploited to learn disentangled representations of numerosity, which can be used to generate novel synthetic images approximately containing a given number of items. Inspired by recent approaches that evaluated the capability of deep generative models to create novel attributes and their combinations [34], we probed the Transformer in different generative scenarios requiring to produce specific numerosities that were never encountered during training. We also analysed the internal structure of the representational code, in order to investigate whether numerosity information could be mapped into a lower dimensional space that preserves the semantics of cardinal numbers [35].

2. Methods

2.1. Problem Formulation

Let $\mathcal{D} = \{(\boldsymbol{x}_1, n_1), \ldots, (\boldsymbol{x}_m, n_m) \text{ s.t. } (\boldsymbol{x}, n) \sim p(\boldsymbol{x}, n),\ n \in \mathcal{N}\}$ be a training dataset consisting of images paired with their respective numerosity (i.e., the number of items contained in each image). The generic $(\boldsymbol{x}, n)$ tuple is sampled i.i.d. from the $p(\boldsymbol{x}, n)$ joint Probability Mass Function (PMF), with $\mathcal{N} \subset \mathbb{N}$. Our goal is to model the $p(\boldsymbol{x}|n)$ conditional PMF, exploiting a density estimation algorithm which relies solely on global attention applied to the raw input images. Modeling such density by disentangling numerosity from other factors of variation should ideally allow to generate images with a controlled number of objects, which could be specified by manipulating the initial state of the generative process[1]. Crucially, the generative model might even learn to produce out-of-distribution samples belonging to areas of the $p(\boldsymbol{x}, n)$ support that are not represented in $\mathcal{D}$, that is, images containing a number of objects that was never experienced during training.

Practically, we focus on an equivalent representation of the target density, which exploits the chain rule to allow the density estimation algorithm to work autoregressively:

$$p(\boldsymbol{x}|n) = \prod_{i=1}^{r} p(x_i|x_1, \ldots, x_{i-1}, n); \tag{1}$$

where $\boldsymbol{x} = (x_1, \ldots, x_r)$ represents a flattened image $\boldsymbol{x}$ made by r pixels. Let $q(\boldsymbol{x}|n, \boldsymbol{\theta}^\star)$ be the approximated conditional PMF produced by the density estimation algorithm, with

[1] Note that although the generative model does not receive explicit knowledge about cardinal numbers during training, the initial state of the generative process is the same for all training images featuring the same numerosity, as explained in Section 2.2.

$\boldsymbol{\theta}^\star$ denoting the optimal model parameters; it originates from the minimization of the following negative log-likelihood:

$$
\begin{aligned}
L(\boldsymbol{\theta}) &= \mathbb{E}_{(\boldsymbol{x},n)\sim\mathcal{D}}\big[-\log q(\boldsymbol{x}|n,\boldsymbol{\theta})\big] && (2)\\
&= \mathbb{E}_{(\boldsymbol{x},n)\sim\mathcal{D}}\left[-\log\prod_{i=1}^{r} q(x_i|x_1,\dots,x_{i-1},n,\boldsymbol{\theta})\right] && (3)\\
&= \mathbb{E}_{(\boldsymbol{x},n)\sim\mathcal{D}}\left[-\sum_{i=1}^{r}\log q(x_i|x_1,\dots,x_{i-1},n,\boldsymbol{\theta})\right]. && (4)
\end{aligned}
$$

Step (4) suggests how to implement the training procedure of our Transformer exploiting PyTorch [36]: we straightforwardly compute the `CrossEntropyLoss` criterion on the model output logits.

2.2. Model Architecture

Our model is an encoder-only Transformer capable of dealing with data characterized by spatial relationships (e.g., images); its backbone, indeed, is built from the self-attention layers devised in [29]. Overall, the following mapping is implemented: $(\boldsymbol{x}, n) \mapsto \boldsymbol{P} \in \mathbb{R}^{q\times p}$; where $\boldsymbol{x} = (x_1,\dots,x_q)$ denotes the categorical input intensities, $q \le r$ and $\boldsymbol{p}_i^T$ (i.e., the i-th row of $\boldsymbol{P}$) represents the conditional PMF[2] associated to x_i.

Input frames are not directly fed into the Transformer encoder: pixel intensities are first scanned following the raster order, and then transformed into learnable embeddings to which the position information is added. Being the positional encodings also learned, it is important to highlight that the model is invariant w.r.t the order in which inputs are supplied; however, once the order is fixed it must be maintained. During the last processing stage, the encoder output goes through a linear layer. Hence, the conditional probability mass functions (PMFs) are computed by applying a softmax function to the produced logits.

The deployed encoder only accepts sequences of real-valued d-dimensional vectors. As a consequence, the supplied dataset entries undergo careful processing. Firstly, the $(x_1,\dots,x_{q-1})$ intensities are mapped into $q-1$ embeddings[3], $\boldsymbol{X} \in \mathbb{R}^{(q-1)\times d}$. Then, the encoder input is computed as:

$$\boldsymbol{H}_0 = \left[\boldsymbol{s}, \boldsymbol{X}^T\right]^T + \boldsymbol{E}; \tag{5}$$

where $\boldsymbol{s} \in \mathbb{R}^d$ encodes the equivalence class to which the considered image belongs (i.e., the numerosity n) while $\boldsymbol{E} \in \mathbb{R}^{q\times d}$ stores information about the pixel positions. Borrowing the machine translation nomenclature, we call $\boldsymbol{s}$ the *Start of String* (SoS). Input embeddings, SoSs and positional encodings are obtained in the same way: the discrete starting values (i.e., intensities, numerosities and positions) trivially become indexes capable of selecting the corresponding rows in one of the $\boldsymbol{W}_{\boldsymbol{X}} \in \mathbb{R}^{p\times d}$, $\boldsymbol{W}_{\boldsymbol{s}} \in \mathbb{R}^{|\mathcal{N}|\times d}$ and $\boldsymbol{W}_{\boldsymbol{E}} \in \mathbb{R}^{r\times d}$ matrices, learned through backpropagation [31–33]. We emphasize that the learned embeddings minimize the introduction of explicit inductive biases: the untrained model, indeed, is completely unaware of the distances between gray levels, the ordering on $\mathbb{N}$ and the correlations of pixels. As a side effect, $\boldsymbol{W}_{\boldsymbol{E}}$ constrains the input image resolution.

[2] The density support, of cardinality p, coincides with the set of input intensities.

[3] Pixel x_q is never consumed by autoregression.

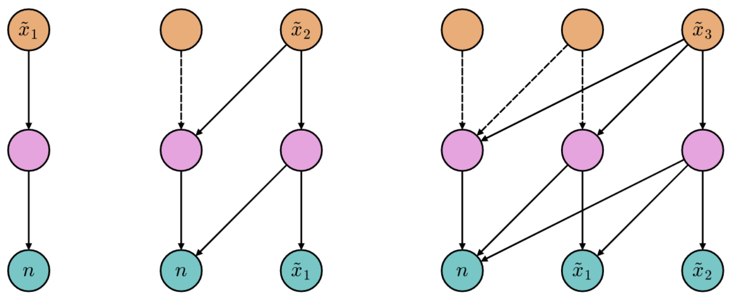

Figure 1. Example *attention graphs* ($L = 2$, $r = 3$) describing the *spontaneous generation* (see Section 2.4) of a novel image. Teal, pink and orange nodes represent the input, hidden and output positions, respectively. For each query node, the outgoing connections indicate which positions can be adaptively weighted. Solid edges denote the active attention flows involved in the generation of the considered gray level. The generative loop starts as in the leftmost graph, where the represented forward pass results in the sampling of the first intensity, $\tilde{x}_1$. In the following graphs, the gray level obtained during the previous pass is appended to the input sequence, and the process is repeated.

The encoder consists of $2L$ properly stacked multi-head scaled dot-product attention (mha$(\bullet)$) and point-wise fully connected (fc$(\bullet)$) sub-layers. Residual connections and layer normalizations (norm$(\bullet)$) complete the architecture. Resuming from (5),

$$\boldsymbol{A}_l = \text{norm}(\boldsymbol{H}_{l-1} + \text{mha}(\boldsymbol{H}_{l-1})) \tag{6}$$
$$\boldsymbol{H}_l = \text{norm}(\boldsymbol{A}_l + \text{fc}(\boldsymbol{A}_l)); \tag{7}$$

describe the encoder pipeline, with the $l \in [1, L]$ subscript denoting the considered layer. The detailed implementations of mha$(\bullet)$, fc$(\bullet)$ and norm$(\bullet)$ can be found in [29]. Finally, the linear$(\bullet)$ and softmax$(\bullet)$ functions are assembled to produce the target conditional densities:

$$\boldsymbol{P} = \text{softmax}(\text{linear}(\boldsymbol{H}_L)). \tag{8}$$

The *attention graphs* [37] showed in Figure 1 help us in explaining how the encoder autoregression is achieved. The represented directed edges identify the allowed attention flows; we masked the missing ones (w.r.t. the respective fully connected, bipartite sub-graphs) to prevent queries from attending to illegal positions. Further details about the model architecture and training hyperparameters are reported in Appendix A.

2.3. Datasets

The Transformer was trained on two different datasets containing images of size 32×32 pixels with a varying number of objects (white dots) placed on a black background. Numerosities were uniformly sampled from the set $\{0, 1, 2, \ldots, 8\}$. Each dataset was split into training, validation and test subsets containing, respectively, 18000, 3600 and 3600 images.

The first dataset, which we call *Uniform Dots*, contained images featuring objects of uniform size (see samples in top row of Fig. 2). In this dataset the numerosity information is perfectly correlated with the total number of active pixels, which does not allow to assess to what extent the Transformer can disentangle numerosity from cumulative area. We thus also introduced a second dataset, which we call *Non-Uniform Dots*, containing images featuring objects of different size and constant (on average) cumulative area (see samples in bottom row of Fig. 2). Let $A_{dot} \sim N(\mu_{dot}, \sigma^2_{dot})$ be the

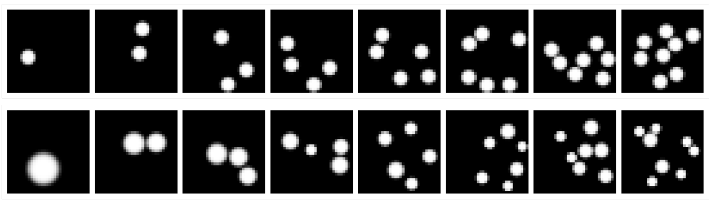
Figure 2. Sample images from the two datasets considered. Stimuli with increasing numerosity (i.e., $\mathcal{N} = \{0, \ldots, 8\}$) are progressively shown from left to right. The top row contains samples from the *Uniform Dots* dataset: the rendered dots have the same radius, do not overlap and are not cut by the boundaries of the boxes. The bottom row contains samples from the *Non-Uniform Dots*, where the area of each dot is sampled from $N(\mu_{frame}/n, \sigma^2_{dot})$.

random variable quantifying the individual area covered by a dot. The total area covered in a frame characterized by n dots can be expressed as:

$$
\begin{aligned}
A_{frame} &= \sum_{i=1}^{n} A_{dot} && (9)\\
&= nA_{dot} && (10)\\
&\sim N(n\mu_{dot}, n^2\sigma^2_{dot}). && (11)
\end{aligned}
$$

Setting $\mu_{dot} = \frac{\mu_{frame}}{n}$ implies that $A_{frame} \sim N(\mu_{frame}, n^2\sigma^2_{dot})$, thus making the expected cumulative area $\mathbb{E}[A_{frame}] = \mu_{frame}$ independent from n[4].

2.4. Generative Tasks

To investigate the emergence of numerosity representations we designed a variety of generation tasks. The objective of these experiments was twofold: on the one hand, they allowed to establish whether the learned representations could be used to produce synthetic images with controlled properties (i.e., featuring a specific numerosity); on the other hand, they allowed to study the internal structure of the Transformer's latent space, in order to investigate whether it could embed the semantics of cardinal numbers.

As an initial assessment, the Transformer was evaluated in a straightforward *conditional generation* task: given the ground-truth $(\boldsymbol{x}, n)$ tuple, the goal is to approximate $\boldsymbol{x}$ through the modeled $q(\boldsymbol{x}|n, \boldsymbol{\theta}^{\star})$, incrementally building the image $\tilde{\boldsymbol{x}}$ according to:

$$\tilde{x}_i = \arg\max_{x} q(x|x_1, \ldots, x_{i-1}, n, \boldsymbol{\theta}^{\star}), \ \forall i \in [1, r]. \tag{12}$$

In other words, each pixel is determined by those preceding it in the fixed scan order, and the current ground-truth pixel values are provided as input at each time step. This task was only used to monitor learning progress, since it is well-known that one-step-ahead prediction is much easier compared to autoregressive self-generation [38].

In the more challenging *spontaneous generation* tasks, we probed the capability of the Transformer to build an entire novel image $\tilde{\boldsymbol{x}}$ from scratch according to:

$$\tilde{x}_i \sim q(x|\tilde{x}_1, \ldots, \tilde{x}_{i-1}, n, \boldsymbol{\theta}^{\star}), \ \forall i \in [1, r]. \tag{13}$$

Unlike (12), each pixel intensity is now conditioned on the previously sampled ones; equation (13), therefore, requires r forward passes for each image. It should be noted that during the first generative step, the encoder input sequence contains only the SoS. Carrying on, the sequence gradually incorporates the new intensity embeddings: $\boldsymbol{s}^T$, $[\boldsymbol{s}, \boldsymbol{X}_1^T]^T$, ..., $[\boldsymbol{s}, \boldsymbol{X}_{r-1}^T]^T$, where the rows of $\boldsymbol{X}_i$ correspond to the first i sampled gray levels. For each SoS considered, a fixed number of 64 images were generated, in order to

[4] In our case, we set $\mu_{frame} = 150$ and $\sigma_{dot} = 8$.

collect statistics about the samples produced. Spontaneous generation was tested under four different conditions:

- ***Spontaneous generation over trained numerosities.*** In this case, the sampling process was initially conditioned on the $|\mathcal{N}|$ numerosity representations learned during the Transformer training (i.e., the rows of $\boldsymbol{W_s}$). That is, the learned SoSs were provided as initial seed.
- ***Spontaneous generation over interpolated numerosities.*** In this case, we tested whether the generative process could be biased toward specific numerosities that were never encountered during training (but nevertheless fell in the training interval) by injecting a novel SoS as initial seed. Defining $\boldsymbol{w}_i^T$ as the row of $\boldsymbol{W_s}$ corresponding to the training numerosity i, the desired conditioning n is injected by simply setting $\boldsymbol{s} = (\boldsymbol{w}_{n-1} + \boldsymbol{w}_{n+1})/2$. In other words, the new representation of n is linearly interpolated from the two closest SoSs.
- ***Spontaneous generation over extrapolated numerosities.*** The generative capability was further pushed by exploring whether the Transformer could be biased to produce numerosities falling outside the training range. Our extrapolation mechanism relies on the *attribute vector* technique described in [39], where the vector representing the direction of change is computed as $\boldsymbol{a} = \boldsymbol{w}_{|\mathcal{N}|} - \boldsymbol{w}_{|\mathcal{N}|-1}$; it represents the direction along which the largest numerosities grow. We conjecture that the representation of a numerosity immediately larger than those included in the training range $[1, \mathcal{N}]$ can be approximated by $\boldsymbol{s} = \boldsymbol{w}_{|\mathcal{N}|} + \alpha\boldsymbol{a}$, for a suitable $\alpha > 0$.
- ***Spontaneous generation with reduced components.*** Although the embedding size is constrained by the encoder architecture, numerosity information might in fact be mapped into a lower-dimensional space, akin to an ordered "number line" [40]. To explore the possibility that the learned SoSs could be arranged along a one- or two-dimensional subspace, we performed a Principal Component Analysis (PCA) on the rows of $\boldsymbol{W_s}$, and used either the 1st or the 1st and 2nd principal components to reconstruct the SoSs used to start the generation process and thus establish whether the sampling quality is affected by such dimensionality reduction.

After all image pixels are generated, the number of dots produced needs to be estimated using a suitable heuristic. For the purpose, we introduced two dataset-specific heuristics. The first counter is a simple area-based heuristic designed to work with uniform dots-like samples. The generated numerosity, indeed, can be computed by simply dividing the area covered by the rendered dots in a frame by the average dot area; such mean value is trivially estimated from the validation split of the dots dataset. Since this heuristic does not work in the case of dots with different size, to estimate the number of dots produced by the Transformer trained on the Non-Uniform Dots dataset we employed a ResNet18 classifier [41], which was trained on a Non-Uniform Dots subset (22000 samples) characterized by $\mathcal{N} = \{0, \ldots, 10\}$. Both counters achieved 100% accuracy on the respective dataset testing splits.

3. Results

After each generation task, we computed the SoS-specific histograms of the generated numerosities. We provide two different histogram visualizations: one depicts the relative frequency of each generated numerosity [34,42], while a 2D histogram is used to reproduce the visualization often used in human behavioral studies [43].

The generation histograms related to the *spontaneous generation over trained numerosities* task are shown in Fig. 3. Especially for the Uniform Dots dataset (top panels), it is evident that the Transformer is able to create synthetic images with a specified numerosity, though the number of generated items is not always accurate. The generation is almost perfect for very small numbers (i.e., 1 and 2), while the model often generates one extra or one fewer item when asked to produce images with larger numerosities (also see sample images reported in Fig. A1). A similar pattern of errors is observed when the Transformer is trained using the Non-Uniform Dots dataset (bottom panels), though in

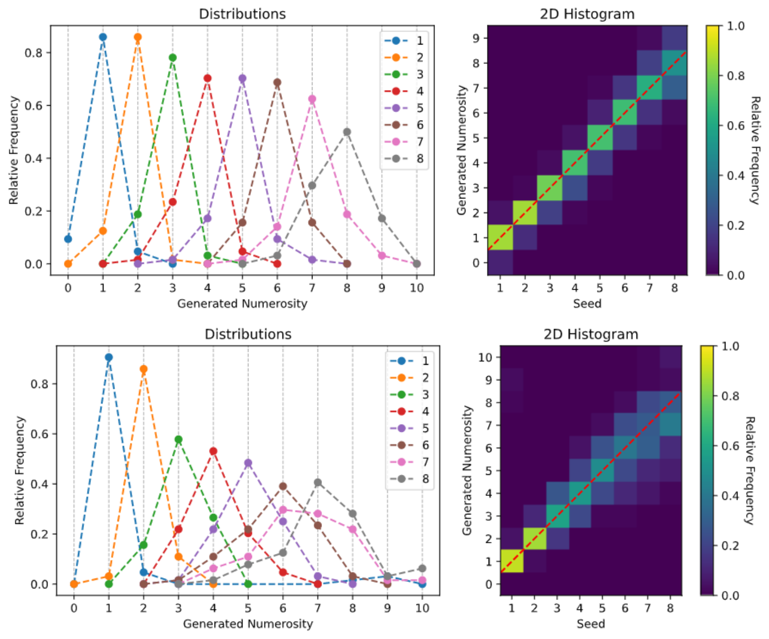

Figure 3. Spontaneous generation over trained numerosities. Top panels: Uniform Dots dataset. Bottom panels: Non-Uniform Dots dataset.

this case the sampling uncertainty associated with larger numerosities increases, and the model sometimes generates images with a mismatch of up to three items. Overall, these results are well-aligned with the existing empirical literature on human behavior, which suggests that numerosity estimates are distributed around the target mean and variability tends to increase with numerosity [42,44], and that numerosity estimation can be altered by confounding non-numerical magnitudes [21,25].

Notably, the synthetic images produced by our Transformer are much more precise compared to samples produced by other deep generative models, such as VAEs or GANs [26,34]. Moreover, differently from previous approaches here we demonstrate that the generation process can be biased toward a specific numerosity, suggesting that attention mechanisms play a key role in allowing a more precise processing of numerosity information.

The generation histograms related to the *spontaneous generation over interpolated numerosities* task are shown in the left panel of Fig. 4. Quite impressively, the Transformer is able to produce images with a specific number of objects even for numerosities that were never encountered during training. For example, by averaging the embeddings corresponding to $n = 1$ and $n = 3$ the model always generates images with exactly 2 dots (yellow line in left panel). An analogous finding holds when interpolating the numerosities $n = 4$ and $n = 6$, though in those cases the number of items is not always perfectly matched (sample images are reported in Fig. A2). These remarkable findings suggest that the emergent representational space approximately encodes the semantics of cardinal numbers, at least within the lower and upper training bounds.

As shown in the right panel of Fig. 4, the results related to the *spontaneous generation over extrapolated numerosities* task further corroborate this hypothesis. Indeed, the attribute vector computed as the difference between the embeddings of the two largest numerosities in the training set (in this case, $n = 4$ and $n = 5$) seems to represent the direction of increase of the numerosity feature: by summing a fraction of such vector to the embedding of $n = 5$, the Transformer can reliably generate images containing 6 items, though sometimes one item appears squeezed or slightly distorted (sample images are reported in Fig. A3). Interestingly, when the attribute vector is scaled by a factor $alpha = 0.5$, the Transformer equally generates images with either 5 or 6 items. However, setting $alpha \geq 2$ did not allow to reliably generate images with 7 items,

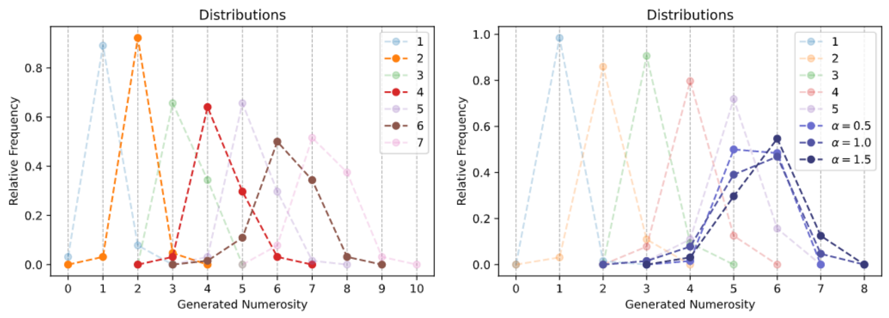

Figure 4. Spontaneous generation over interpolated (left) and extrapolated (right) numerosities. Trained numerosities are represented by semi-transparent curves, while solid curves represent unseen numerosities.

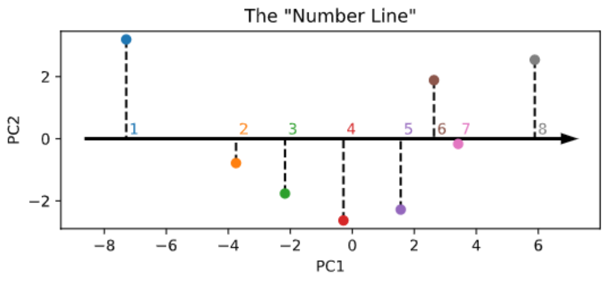

Figure 5. Visualization of the lower-dimensional embedding space resulting from PCA.

suggesting that the learned embeddings approximately capture a sort of "successor function" only over the local neighborhood of a specific numerosity.

The lower-dimensional manifold structure of the encoder space is shown in Fig. 5. Interestingly, and in partial alignment with other recent computational work [35], it seems that the topology of the numerosity embeddings preserves the strict ordering of cardinal numbers, even though the Transformer did not explicitly receive such information during training. This is evident even by just looking at the first principal component (x-axis in the figure), which suggests that numerosity information could be internally organized as a one-dimensional "number line" [20,45]. However, differently from [35] here we find that the second principal component does not monotonically encode cardinal information, but suggests a periodic pattern. As a control analysis, in Fig. A4 we also show the PCA projection resulting right after the random initialization of the embeddings, which indeed does not reflect any ordering structure.

The results related to the *spontaneous generation with reduced components* task suggest that when the embeddings are projected into such lower-dimensional manifold the generative abilities of the model are preserved: as shown in the top panels of Fig. 6, the Transformer can generate samples with remarkable accuracy even when only the first principal component is retained. Adding the second principal component (bottom panels of Fig. 6) allows to further improve the generation precision, though numerosities mapped to nearby points in the lower dimensional space (i.e., $n = 6$ and $n = 7$) are frequently confounded.

4. Conclusions

In this work we investigated whether state-of-the-art deep learning architectures based on attention mechanisms could learn disentangled representations of numerosity from a set of images containing a variable number of items. Our simulations not only show that Transformers can successfully learn to generate synthetic images featuring a target numerosity, but also that they can interpolate and extrapolate the generation process to previously unseen numerosities. These remarkable findings suggest that Transformers can indeed disentangle numerosity from other non-numerical visual fea-

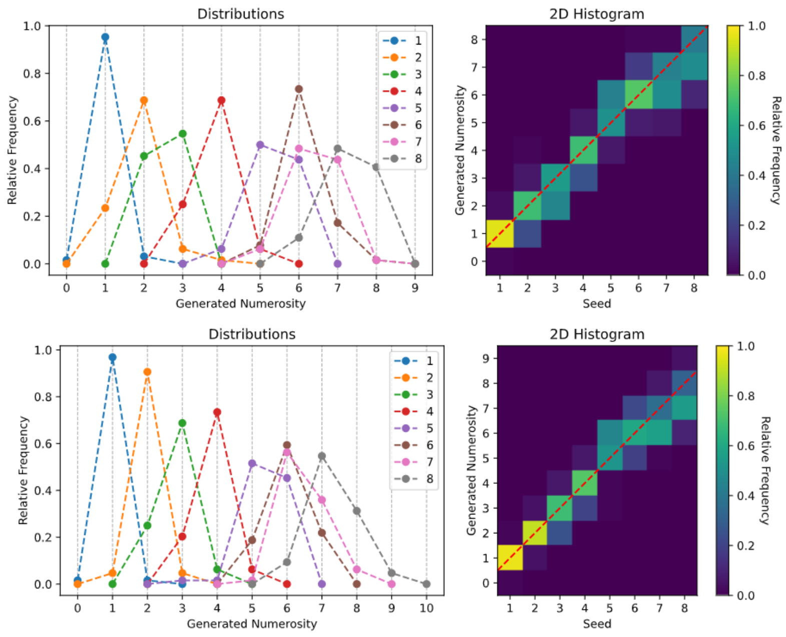

Figure 6. Spontaneous generation with reduced components, considering either just 1 principal component (top panels) or 2 principal components (bottom panels).

tures. However, it should be stressed that the generation process is error-prone and thus reflects an *approximate* representation of numerical information. Moreover, though we are impressed by the Transformer's generative capabilities, in real world scenarios the number of training patterns can be exponentially smaller than the support of the probability mass function to be estimated, which makes generalization to out-of-distribution samples particularly challenging [34]. A key open issue is thus to establish whether domain-general deep learning architectures could extrapolate numerical knowledge well beyond the limit of their training distribution, which would require to learn more abstract conceptual structures, such as the successor function [46], that form the foundation of our understanding of natural numbers [47].

Another limitation of Transformer architectures is related to their computational complexity: naive implementations have a quadratic cost in the number of pixels in terms of both memory and computation, preventing their scaling to high resolutions. Recent work attempted to mitigate this issue by approximating global attention in different ways, for example by restricting self-attention receptive fields to local neighborhoods [31], reducing image resolution [32] or focusing on image patches [33]. In the present work, the size of our images allowed to efficiently train and test the Transformer architecture; however, future work should better clarify whether more effective attention mechanisms could be employed to scale-up the model to realistic image sizes.

Author Contributions: Conceptualization and supervision, A.T. and M.Z.; methodology, T.B. and A.T.; software, simulations, analysis and visualization, T.B.; writing—original draft preparation, T.B. and A.T.; writing—review and editing, T.B., A.T. and M.Z.; project administration, M.Z.; funding acquisition, A.T. and M.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Cariparo Excellence Grant 2017 (project "Numsense").

Data Availability Statement: The source code for the simulations will be available to download at https://github.com/BoCtrl-C?tab=repositories.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Appendix A. Model Hyperparameters and Supplementary Figures

Our Transformer is available in three different sizes: "S" ($\sim$ 37000 parameters), "M" ($\sim$ 136000 parameters) and "L" ($\sim$ 568000 parameters); Table A1 reports the corresponding hyperparameters. For publication purposes, all the presented results refer to the size "M" model. However, we also investigated the performance of the small and large variants on a subset of the introduced tasks, without noticing major differences. All models were trained using Adam optimizer [48] ($\beta_1 = 0.9$, $\beta_2 = 0.999$) with a batch size of 16. The training always stops when the validation loss does not improve, w.r.t. the best loss obtained, for 5 epochs in a row. We set the initial learning rate to 0.003 after a search in the set $\{0.03, 0.01, 0.003, 0.001\}$. Furthermore, the learning rate decays by 0.1 every 25 training epochs. On the workstation exploited for our simulations (equipped with an NVidia GTX 1080 graphic card) each training session took, on average, 1 hour and 45 minutes. To reduce the Transformer computational demand, we pre-processed the dataset entries by uniformly quantizing (16 levels) the input intensities (i.e., $p = 16$).

Table A1. Hyperparameters characterizing the available model sizes. See the PyTorch Transformer documentation for more details about `nhead` and `dim_feedforward` (https://pytorch.org/docs/stable/generated/torch.nn.Transformer.html).

Hyperparameter	**Value**		
	Size "S"	Size "M"	Size "L"
`d_model` (d)	16	32	64
`nhead`	2	2	4
`num_encoder_layers` (L)	6	8	10
`dim_feedforward`	64	128	256

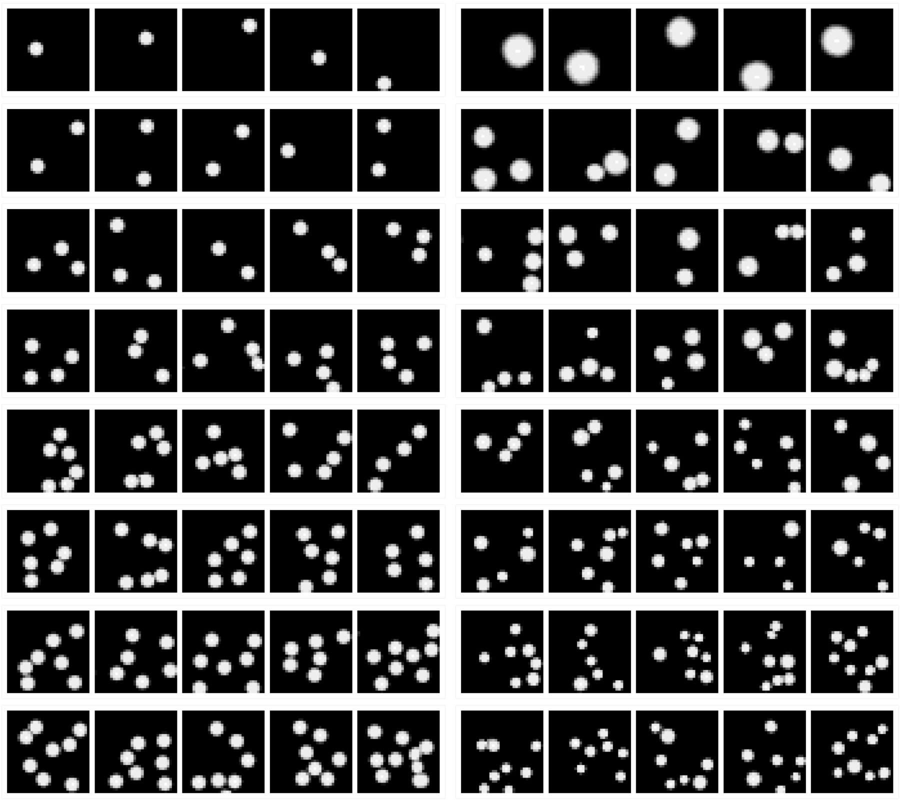

Figure A1. Sample images produced in the *spontaneous generation over trained numerosities* task. Left panel: *Uniform Dots*. Right panel: *Non-Uniform Dots*. Images produced with increasing generation seeds (i.e., $\mathcal{N} = \{0, \ldots, 8\}$) are progressively shown from top to bottom. Sample images with a number of dots that does not match the seed are purposely included for illustration.

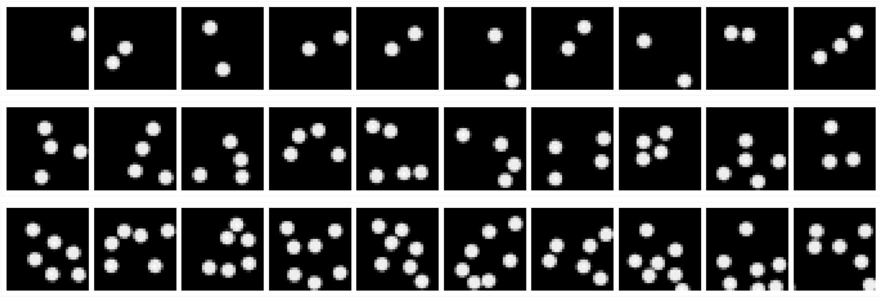

Figure A2. Sample images conditioned on the interpolated SoSs (i.e., *spontaneous generation over interpolated numerosities* task). From top to bottom, the displayed rows correspond to the unseen numerosities 2, 4 and 6, respectively.

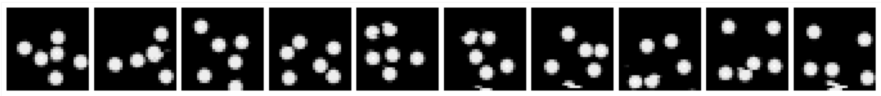

Figure A3. Sample images conditioned on the extrapolated SoS (i.e., *spontaneous generation over extrapolated numerosities* task). The results reported refer to $\alpha = 1$. Note how the rendering of dots is qualitatively less precise than the ones shown in Figures A1 and A2.

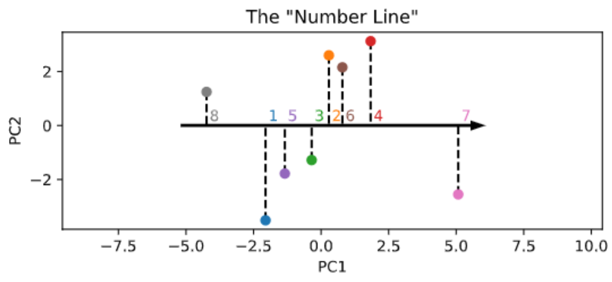

Figure A4. Visualization of the lower-dimensional embedding space right after random initialization.

References

1. Bengio, Y.; Courville, A.; Vincent, P. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* **2013**, *35*, 1798–1828.
2. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *nature* **2015**, *521*, 436–444.
3. Zorzi, M.; Testolin, A.; Stoianov, I.P. Modeling language and cognition with deep unsupervised learning: a tutorial overview. *Frontiers in psychology* **2013**, *4*, 515.
4. Zhuang, C.; Yan, S.; Nayebi, A.; Schrimpf, M.; Frank, M.C.; DiCarlo, J.J.; Yamins, D.L. Unsupervised neural network models of the ventral visual stream. *Proceedings of the National Academy of Sciences* **2021**, *118*.
5. Clark, A. Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behavioral and brain sciences* **2013**, *36*, 181–204.
6. Friston, K. The free-energy principle: a unified brain theory? *Nature reviews neuroscience* **2010**, *11*, 127–138.
7. Hinton, G.E. Learning multiple layers of representation. *Trends in cognitive sciences* **2007**, *11*, 428–434.
8. Testolin, A.; Zorzi, M. Probabilistic models and generative neural networks: Towards an unified framework for modeling normal and impaired neurocognitive functions. *Frontiers in Computational Neuroscience* **2016**, *10*, 73.
9. Hinton, G.E.; Osindero, S.; Teh, Y.W. A fast learning algorithm for deep belief nets. *Neural computation* **2006**, *18*, 1527–1554.
10. Le, Q.V. Building high-level features using large scale unsupervised learning. 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, 2013, pp. 8595–8598.
11. Sadeghi, Z.; Testolin, A. Learning representation hierarchies by sharing visual features: a computational investigation of Persian character recognition with unsupervised deep learning. *Cognitive processing* **2017**, *18*, 273–284.
12. Testolin, A.; Stoianov, I.; Zorzi, M. Letter perception emerges from unsupervised deep learning and recycling of natural image features. *Nature human behaviour* **2017**, *1*, 657–664.
13. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* **2013**.
14. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *arXiv preprint arXiv:1406.2661* **2014**.
15. Karras, T.; Laine, S.; Aila, T. A style-based generator architecture for generative adversarial networks. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4401–4410.
16. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. Proceedings of the IEEE international conference on computer vision, 2017, pp. 2223–2232.
17. Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework **2016**.
18. Chen, X.; Duan, Y.; Houthooft, R.; Schulman, J.; Sutskever, I.; Abbeel, P. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv preprint arXiv:1606.03657* **2016**.
19. Burgess, C.P.; Matthey, L.; Watters, N.; Kabra, R.; Higgins, I.; Botvinick, M.; Lerchner, A. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390* **2019**.
20. Dehaene, S. *The number sense: How the mind creates mathematics*; OUP USA, 2011.
21. Gebuis, T.; Reynvoet, B. The interplay between nonsymbolic number and its continuous visual properties. *Journal of Experimental Psychology: General* **2012**, *141*, 642.
22. Stoianov, I.; Zorzi, M. Emergence of a'visual number sense'in hierarchical generative models. *Nature neuroscience* **2012**, *15*, 194–196.
23. Testolin, A.; Zou, W.Y.; McClelland, J.L. Numerosity discrimination in deep neural networks: Initial competence, developmental refinement and experience statistics. *Developmental science* **2020**, *23*, e12940.
24. Zorzi, M.; Testolin, A. An emergentist perspective on the origin of number sense. *Philosophical Transactions of the Royal Society B: Biological Sciences* **2018**, *373*, 20170043.
25. Testolin, A.; Dolfi, S.; Rochus, M.; Zorzi, M. Visual sense of number vs. sense of magnitude in humans and machines. *Scientific reports* **2020**, *10*, 1–13.
26. Zanetti, A.; Testolin, A.; Zorzi, M.; Wawrzynski, P. Numerosity Representation in InfoGAN: An Empirical Study. International Work-Conference on Artificial Neural Networks. Springer, 2019, pp. 49–60.
27. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings; Bengio, Y.; LeCun, Y., Eds., 2015.
28. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to Sequence Learning with Neural Networks. Advances in Neural Information Processing Systems; Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N.; Weinberger, K.Q., Eds. Curran Associates, Inc., 2014, Vol. 27.
29. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.u.; Polosukhin, I. Attention is All you Need. Advances in Neural Information Processing Systems; Guyon, I.; Luxburg, U.V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; Garnett, R., Eds. Curran Associates, Inc., 2017, Vol. 30.
30. Ramsauer, H.; Schäfl, B.; Lehner, J.; Seidl, P.; Widrich, M.; Gruber, L.; Holzleitner, M.; Pavlović, M.; Sandve, G.K.; Greiff, V.; others. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217* **2020**.

31. Parmar, N.; Vaswani, A.; Uszkoreit, J.; Kaiser, L.; Shazeer, N.; Ku, A.; Tran, D. Image Transformer. Proceedings of the 35th International Conference on Machine Learning; Dy, J.; Krause, A., Eds. PMLR, 2018, Vol. 80, *Proceedings of Machine Learning Research*, pp. 4055–4064.
32. Chen, M.; Radford, A.; Child, R.; Wu, J.; Jun, H.; Luan, D.; Sutskever, I. Generative Pretraining From Pixels. Proceedings of the 37th International Conference on Machine Learning; III, H.D.; Singh, A., Eds. PMLR, 2020, Vol. 119, *Proceedings of Machine Learning Research*, pp. 1691–1703.
33. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; Houlsby, N. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, 2020, [arXiv:cs.CV/2010.11929].
34. Zhao, S.; Ren, H.; Yuan, A.; Song, J.; Goodman, N.; Ermon, S. Bias and Generalization in Deep Generative Models: An Empirical Study. Advances in Neural Information Processing Systems; Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; Garnett, R., Eds. Curran Associates, Inc., 2018, Vol. 31.
35. Kondapaneni, N.; Perona, P. A Number Sense as an Emergent Property of the Manipulating Brain. *arXiv preprint arXiv:2012.04132* **2020**.
36. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems; Wallach, H.; Larochelle, H.; Beygelzimer, A.; dAlché-Buc, F.; Fox, E.; Garnett, R., Eds. Curran Associates, Inc., 2019, Vol. 32.
37. Abnar, S.; Zuidema, W. Quantifying Attention Flow in Transformers, 2020, [arXiv:cs.LG/2005.00928].
38. Cenzato, A.; Testolin, A.; Zorzi, M. Long-Term Prediction of Physical Interactions: A Challenge for Deep Generative Models. International Conference on Machine Learning, Optimization, and Data Science. Springer, 2020, pp. 83–94.
39. Carter, S.; Nielsen, M. Using Artificial Intelligence to Augment Human Intelligence. *Distill* **2017**. https://distill.pub/2017/aia, doi:10.23915/distill.00009.
40. Dehaene, S. The neural basis of the Weber–Fechner law: a logarithmic mental number line. *Trends in cognitive sciences* **2003**, *7*, 145–147.
41. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778. doi:10.1109/CVPR.2016.90.
42. Sella, F.; Berteletti, I.; Lucangeli, D.; Zorzi, M. Spontaneous non-verbal counting in toddlers. *Developmental science* **2016**, *19*, 329–337.
43. Revkin, S.K.; Piazza, M.; Izard, V.; Cohen, L.; Dehaene, S. Does subitizing reflect numerical estimation? *Psychological science* **2008**, *19*, 607–614.
44. Testolin, A.; McClelland, J.L. Do estimates of numerosity really adhere to Weber's law? A reexamination of two case studies. *Psychonomic Bulletin & Review* **2021**, *28*, 158–168.
45. Harvey, B.M.; Klein, B.P.; Petridou, N.; Dumoulin, S.O. Topographic representation of numerosity in the human parietal cortex. *Science* **2013**, *341*, 1123–1126.
46. Leslie, A.M.; Gelman, R.; Gallistel, C. The generative basis of natural number concepts. *Trends in cognitive sciences* **2008**, *12*, 213–218.
47. Testolin, A. The challenge of modeling the acquisition of mathematical concepts. *Frontiers in human neuroscience* **2020**, *14*.
48. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings; Bengio, Y.; LeCun, Y., Eds., 2015.