Being Creative: A Cross-Domain Mapping Network

Jichen Wu, Maarten H. Lamers, Wojtek J. Kowalczyk
Leiden Institute of Advanced Computer Science
Leiden University
Leiden
j.wu.13@umail.leidenuniv.nl, m.h.lamers@liacs.leidenuniv.nl, w.j.kowalczyk@liacs.leidenuniv.nl

Abstract

The ability to extract features from objects or concepts and to connect them in a meaningful way is believed to be crucial to creativity. This paper proposes a novel computational model for creative behaviour, by learning extractions and connections separately. Such separation enables adaptive feature and object connections, which means one object can be connected to different other objects within different contexts. This paper applies recent cognitive theories of computational creativity to two specific tasks: an image-to-image mapping and music-to-video mapping. In both cases deep neural networks are used to automate these two creative cognition tasks.

Introduction

It is not easy to answer the question: what is creativity? Nevertheless, many agree that the ability to make connections or combinations between concepts is a key component of creativity. Mekern, Hommel, and Sjoerds (2019) implied a unitary model of creative cognition. This model, which is inspired by Hommel and Wiers (2017), is based on adaptive relations between features of concepts and ideas. "Features of concepts" means that the relations are not directly made between concepts but through features, and that one concept is represented by many independent features. "Adaptive" implies the weighting of possible features according to different situations so that the contributions of different features vary to adapt to the current situation. This unitary creative model is arguably an integration and simplification of the divergent creativity model (Kenett et al. 2018) and the convergent creativity model (Kajić et al. 2017).

On the practical side, however, few computational models that stress features and their relations were built into machines. Some previous studies did partly model these properties: Olteţeanu and Falomir (2016) used an "Object Replacement and Composition" system, in which the features of an object seem to have to be manually decided and the size of the dataset is relatively small. This approach makes the dataset accurate and human-understandable. However, as they pointed out, the system needs a larger dataset or a different approach to build the feature space to perform on a larger scope. In another creative machine (Augello et al. 2016), features of an image object are extracted directly based on color and texture information. Although this approach avoided manually building datasets, it took the risk of losing much information, for example, distribution of color, or shapes within the image. These issues reveal the need for a more general and automatic method to efficiently acquire more comprehensive features and relations.

Luckily, the development of machine learning makes it possible to extract various kinds of useful features from large datasets. Therefore, given two domains X and Y, we can extract features from points in these domains into feature spaces Z_X and Z_Y and consider the problem of finding various mapping functions that map Z_X onto Z_Y that satisfy some context dependent constraints. We should notice here that, as we explain in Related Work, the term 'feature' has a somewhat different meaning in the context of 'machine learning' and 'cognition research'. Nevertheless, without any risk of confusion we will use this term without specifying the context.

Previous studies (Augello et al. 2016; Olteţeanu and Falomir 2016; Huang et al. 2018; Liu, Breuel, and Kautz 2017) can only find a single mapping function because this function relies on some level of equality between the feature vectors. For example, a red, thick T-shirt should be mapped to red, thick trousers. However, humans do not always relate things by similarity of features. For example, we might think a red, thin T-shirt makes a good pair with blue, thick trousers because of the influence of fashion trends or because they are similarly rare in their domains. To find multiple mapping functions we propose two criteria. First, if humans experience two things together for several times, they may naturally connect them. So if previously experienced pairs from X and Y are provided, a mapping function should map their feature vectors together. This criterion is named: previously experienced mapping. Second, similar objects in domain X should be mapped to similar objects in domain Y, while dissimilar objects in X should be mapped to dissimilar objects in Y. This criterion is named: topology mapping.

While a few previous models (Huang et al. 2018; Zhu et al. 2017b) can already learn non-deterministic mappings between domains, they still cannot learn different mappings based on different criteria. Rather the mapped data points are randomly selected based on statistic distributions. Our research serves as a realization for recent creative cognition models as well as an exploration of creativity in contemporary deep learning models. It aims to connect cognition

Proceedings of the 11th International Conference on Computational Creativity (ICCC'20)
ISBN: 978-989-54160-2-8

theory and computational applications.

In the next section, related work regarding the use of features in computational creativity theories and its relevance in computational technologies is introduced. The methodology and framework of our Cross-Domain Mapping Network (CDMN) are described in *Method*, followed by experiments assessing the effectiveness and creative behaviours of the CDMN. Finally, conclusions are provided.

Related Work

Related work is discussed in four parts. First, the trend of considering features in creative cognition is introduced. Secondly, machine learning methods for feature encoding are briefly described. Third, methods for constructing feature connections are recognized from the field of machine learning, specifically image-to-image mapping networks. Finally, possibilities for multi-modal translations (audio, image, video) are provided.

Features and Computational Creativity

A feature set is a distributed representation of a concept, where 'distributed' means a feature set is maximally representative of the concept on a certain level of significance. Feature extraction and feature connections have an important role in the theory of computational creativity since the distinction between convergent and divergent thinking of creativity (Guilford 1967). Divergent thinking is to generate creative ideas or solving problems creatively by exploring many possible solutions whereas convergent thinking is to provide a single best solution to a well-defined question. The ability to make associations is believed to be important to both processes (Gabora 2010; Mednick 1962).

Both divergent and convergent thinking are modeled explicitly in a dual-process computational painter by Augello et al. (2016). The key operation in this painter is to replace image part A with image B that shares features such as color and texture. However, in such dual-process models, outcomes of divergent and convergent processes are often hard to distinguish and outcomes of each process rely to some extent on the interplay between these two processes (Mekern, Hommel, and Sjoerds 2019)

Merken et al.(2019) proposed a unitary model for creative behaviour in which creative behaviours are facilitated by the interplay between features. To achieve a degree of flexibility, under a different context different connections between features are activated. From Merken's unitary model, we identify two processes: (1) the encoding of distributed features, and (2) flexible connections between features to facilitate contextualization and individual differences. Along with these two processes, three criteria are proposed to evaluate a creative model: (1) features are distributed and representative, (2) the connections are flexible under different circumstances, and (3) individual differences (flexible and persistent) can be modeled.

Encoding and Decoding Features

Although the importance of features is identified, the computational creativity community still does not have many reliable and automatic ways to encode (extract) features from concepts. Previous programs either involve manual encoding (e.g. (Olteţeanu and Falomir 2016)) or the encoded features are not well-distributed (a concept cannot be fully represented by its feature set; e.g. (Augello et al. 2016)). Fortunately, neural networks have been developed to find complex features from raw data. A prominent example of such networks is the autoencoder (e.g. (Hinton and Salakhutdinov 2006)). One may think that the features encoded by an autoencoder are just numbers and thus of a different nature from 'color', 'shape' or 'texture'. However, what makes features crucial is not whether they are abstract numbers, activation of neurons or concrete attributes, but whether they serve as a set of distributed representations of a concept or an object. The encoding-decoding process of an autoencoder ensures that the encoded features are maximally representative of the input data.

One property of an autoencoder is that the encoded features are data-specific. With a dataset of cup images, it can never recognize 'concave' as a feature because it is not distinctive. Because of this, creative problem solving as by Olteţeanu and Falomir (2016) cannot be easily achieved without an extensive dataset. However, this property also enables autoencoders to generate never-before-seen objects from features. Utilizing this advantage, this paper aims at a generative model instead of an associative model.

Feature Connection

Several recent studies regarding Domain Transfer Networks (DTNs) have already achieved the process of feature encoding and connection to some extent. However, relevance between DTNs and creativity theories is hardly mentioned. Thus, in this section, the relevance of DTNs in the connecting of features is identified.

DTNs have been studied for image-to-image translation. Early works (Yoo et al. 2016; Zhu et al. 2017a) do not perform any manipulations on the encoded features. Some successors (Zhu et al. 2017b; Taigman, Polyak, and Wolf 2017) found that having features as inputs to the generators enhanced the performance of the network. However, in these methods, features from different domains are not explicitly connected.

Liu et al. (2017) were the first to model explicitly the connection of features, although this connection simply equates features of X with features of Y without flexibility of such connections. Huang et al. (2018) proposed an improvement that provides some degree of flexibility of connections. It implies that some features are always connected and other features are never connected. Since true flexibility is deciding which features are relevant to the context and should be connected adaptively, our work makes an attempt to implement such flexibility.

Cross-modal Temporal Data Generation

We are interested not only in the domain of images. The problem becomes more tricky if the domains include temporal data, such as video or music, or require cross-modal (audio to visual, visual to audio) mapping. With other applications showing the possibilities (Song et al. 2019;

Ephrat and Peleg 2017), we want to explore and evaluate the creative behaviour in audio-video translation also.

Method

The two steps of creative cognition (feature encoding and flexible feature connecting) are implemented in a computer program. Although the programs are specially designed for image data and temporal data, the methodology should in principle be suitable for other types of data as well.

Mapping Functions

A mapping function m is, in principle, a function that translates each data point z_x in feature space Z_X to a data point $z_{x\to y}$ in feature space Z_Y: $z_{x\to y} = m(z_x)$. Since usually there are infinitely many points in a feature space, the number of mapping functions is infinite. To attack this problem we first cluster Z_X and Z_Y into n_X and n_Y clusters and construct mapping functions to map clusters upon each other. Thus a mapping function from Z_X to Z_Y exists among the finite set of possibilities with size $(n_Y)^{n_X}$.

However much information could be lost if the feature vector only carries clustering information. To overcome this, a feature vector z_x is split into two vectors: a cluster vector c_x in a finite space and a vector carrying other detail information v_x in an infinite space. Next, the feature extraction function E_X is defined: $c_x, v_x = E_X(x)$. Similarly we have E_Y. While there are $(n_Y)^{n_X}$ possible mappings from the cluster vector space of X to the cluster vector space of Y, v_x is passed unchanged: $v_{x\to y} = v_x$. The relation between c_x and v_x can be understood using Fig. 1. While c_x defines the center of a cluster in the space, v_x is a small vector deviating from this center. With clusters numbers $n_x = n_y = 1$, this model is identical to the shared latent space assumption proposed by Liu et al. (2017). With $n_x \to \infty$, $n_y \to \infty$ and $v \to 0$, in theory it is possible to construct any arbitrary mapping function. The process of mapping a data point x to domain Y is:

$$c_x, v_x = E_X(x)$$
$$c_{x\to y} = m(c_x)$$
$$y = G_Y(c_{x\to y}, v_x)$$

where G_Y is a function that decodes a feature vector back into the domain Y, or $\hat{y} = G_Y(E_Y(y))$ where $\hat{y} \approx y$.

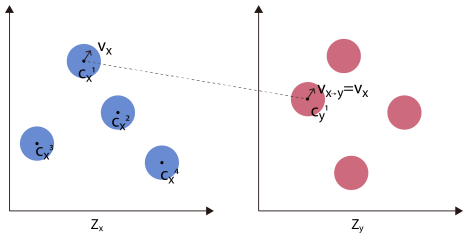

Figure 1: Relations between c and v and how a mapping function works.

To find good mapping functions we apply two criteria mentioned in the Introduction: previously experienced mapping and topology mapping. First, when previously experienced pairs $\{(x_1, y_1), ..., (x_{nn}, y_{nn})\}$ are present, assuming that $c_{x_i}, v_{x_i} = E_X(x_i)$, the loss of a mapping function can be measured by how well it matches clusters of given pairs:

$$\mathcal{L}_{pair} = \sum_{i=1}^{nn} w_i \cdot \text{eval}(m(c_{x_i}), c_{y_i})$$

where $\text{eval}(m(c_{x_i}), c_{y_i})$ returns 0 if $m(c_{x_i}) = c_{y_i}$, 1 otherwise, and w_i is a weight assigned to each pair i.

Second, when topology mapping is used, we first assume that the topology is preserved with the feature extraction functions E_X and E_Y so similar objects have small distance in feature space. The next loss function measures how well a mapping function m preserves the distances between clusters of x:

$$\mathcal{L}_{topo} = \sum_{i=1}^{n_X}\sum_{j=1}^{n_X}(d(c_x^i, c_x^j) - d(m(c_x^i), m(c_x^j)))^2$$

where c_x^i represents the i-th cluster of the total n_X clusters and $d(c_x^i, c_x^j) \in [0, 1]$ is the normalized Euclidean distance between c_x^i and c_x^j. $\mathcal{L}_{topo}$ is called the stress function in multidimensional scaling.

Given weights w_{topo}, the overall loss to be minimized is:

$$\mathcal{L}_{map} = \mathcal{L}_{pair} + w_{topo}\mathcal{L}_{topo}$$

An algorithm to find good solutions of $\mathcal{L}_{map}$ is subject to the criteria that the mapping functions should be able to model individual differences. In this paper genetic algorithms are used because their design can facilitate the modeling of flexible and persistent (explorative and exploitative) individuals.

Network for Feature Encoding

The functions E_X, G_X, E_Y, G_Y are learned by a neural network. We assume that c_x, v_x, c_y, v_y are of the same dimensionality. Furthermore, c_x, c_y can be represented by one-hot vectors h_x, h_y:

$$c_x = H_X(h_x), \quad c_y = H_Y(h_y)$$

We update E_X so that: $h_x, v_x = E_X(x)$ and update m so that $h_{x\to y} = m(h_x)$ (which does not change the functionality of E_X and m). The same change is also made to E_Y. Makhzani et al. (2015) have shown that this way it is possible for the encoder to learn cluster representations via one-hot vectors.

The complete structure of the network is shown in Figure 2. It has two functions. When passing c_x, v_x to G_X, it is an autoencoder to reconstruct x, when passing $H_Y(m(h_x)), v_x$ to G_Y, it is a mapping network. For networks E and G, similar structures to Liu et al. (2017) are used.

Training for autoencoding The two autoencoder structures are trained independently. For autoencoder (E_X, H_X, G_X), we want reconstructions $\hat{x}$ to approach inputs x. Here L1 loss is used:

$$\mathcal{L}^x_{recon} = E[||x - \hat{x}||_1]$$

where $\hat{x} = G_X(H_X(h_x), v_x)$ and $h_x, v_x = E_X(x)$. v_x and v_y need to follow the same distribution for the mapping to work. We let them both follow the standard distribution

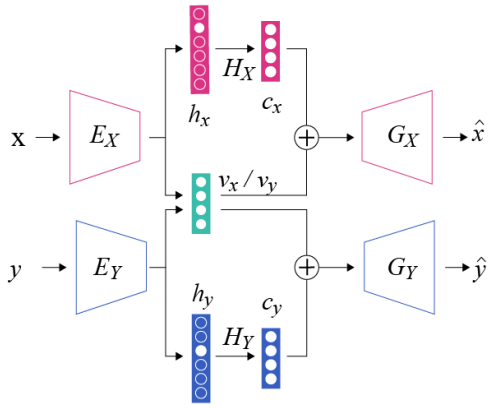

Figure 2: Complete structure of CDMN.

$\mathcal{N}(0, I)$ where I is the identity matrix. This is done by using a VAE structure (Kingma and Welling 2014) that uses KL-divergence loss:

$$\mathcal{L}^x_{KL} = KL(E_X(x)[1]||N(0, I))$$

where $E_X(x)[1] = v_x$ and h_x is expected to be a one-hot vector representing unsupervised clustering information. Adversarial training is used with a discriminator D_X that tries to tell h_x from a random real one-hot vector h^r_x of the same dimensionality, resulting in loss function:

$$\mathcal{L}^x_{adve} = E[log(1 - D_X(E_X(x)[0]))] + E[log(D_x(h^r_x))]$$

where $E_X(x)[0] = h_x$. An illustration of the training process is shown in Figure 3.

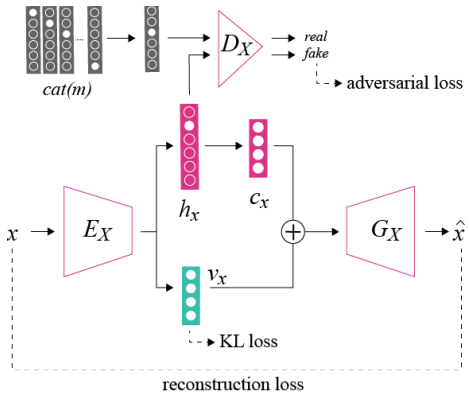

Figure 3: Network structure at training for autoencoding loss.

Similarly, we have loss functions for y, and a total loss

$$\mathcal{L}^{ae}_{total} = w_{recon} \cdot (\mathcal{L}^x_{recon} + \mathcal{L}^y_{recon}) + w_{kl} \cdot (\mathcal{L}^x_{KL} + \mathcal{L}^y_{KL}) + w_{adve} \cdot (\mathcal{L}^x_{adve} + \mathcal{L}^y_{adve})$$

which is minimized by E, H and G while maximized by D.

Training for mapping There are two problems if the network is only trained minimizing L^{ae}_{total}. First, even though L_{KL} penalizes v_x, v_y that do not follow the Gaussian distribution, they nonetheless tend to deviate (Makhzani et al. 2015), especially in high dimensional spaces. This leads to the situation that v_x and v_y follow different distributions and the mapping $G_Y(c_y, v_x)$ will only generate a noisy output. Second, the clustering has strong bias — there could be one cluster containing half of the training set while most other clusters are empty. To overcome these issues, a joint training process is designed. With v_x and a random vector h^r_y, $G_Y(H_Y(h^r_y), v_x)$ learns to generate a realistic image with the help of a discriminator D^{img}_Y for generated outputs:

$$\mathcal{L}^y_{GAN} = E[log(1 - D^{img}_Y(G_Y(H_Y(h^r_y), E_X(x)[1]))] + E[log(D^{img}_Y(y))]$$

This ensures that G_Y learns the distribution of v_x and also the full distribution h_y. Cycle consistence loss (Zhu et al. 2017a) is also used:

$$\mathcal{L}^y_{cyc} = E[||v_x, h^r_y - E_Y(G_Y(H_Y(h^r_y) + v_x))||_1]$$

where $v_x = E_X(x)[1]$. These processes are illustrated in Figure 4.

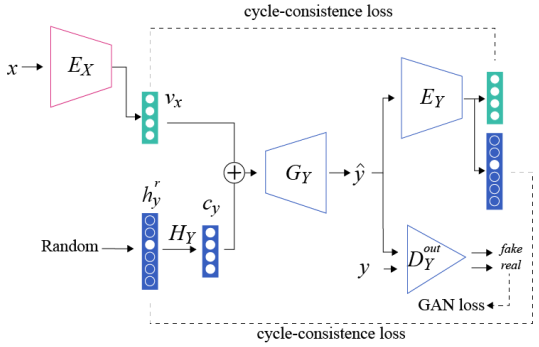

Figure 4: Network structure and training for mapping loss.

Similarly, we have the loss for $y \rightarrow x$ and a total loss of:

$$\mathcal{L}^{map}_{total} = w_{GAN} \cdot (\mathcal{L}^x_{GAN} + \mathcal{L}^y_{GAN}) + w_{cyc} \cdot (\mathcal{L}^x_{cyc} + \mathcal{L}^y_{cyc})$$

which is minimized by E, H and G while being maximized by D. During the training process, the network is trained on $\mathcal{L}^{ae}_{total}$ and $\mathcal{L}^{map}_{total}$ iteratively.

Network for cross-modal temporal data Besides unimodal image-to-image translation tasks, a creative model should also be able to solve cross-modal non-static (temporal) translation tasks. Such tasks post more restrictions on the autoencoder networks. A variant of the network shown in Figure 2 is specifically designed for audio-to-video translation with two generators consisting of LSTM convolutional blocks (Xingjian et al. 2015). The new functions are named G^m_X and G^m_Y: $\hat{x}, mem^{t+1}_x = G^m_X(c_x, v_x, mem^t_x)$ where mem^t_x is the memory of the network G^m_X at time t.

Experiments

This section is divided into two parts. First, configurations and technical properties of the model are studied. Then, scenarios are provided to study the model's creative behaviour.

Proceedings of the 11th International Conference on Computational Creativity (ICCC'20)
ISBN: 978-989-54160-2-8

Performance Analysis

This section studies configurations and technical properties of (1) an image-to-image mapping model, and (2) a music-to-video mapping model. The results are mostly qualitative as quantitative results can hardly be captured and evaluated.

Image-to-image mapping The evaluation is performed using a dataset of clothing images that were obtained from the Alibaba Tianchi Big Data Competition[1] entitled 'Keypoints Detection of Apparel – Challenge the Baseline' and converted to 64x64 pixels. Images of blouses ($n \approx 5000$) are used for domain X, trousers and skirts ($n \approx 15000$) are used for domain Y.

First, the network's sensitivity to hyperparameter settings is evaluated. Testing for distributions of v (either $N(0, I)$ or $N(0, 0.1I)$), decoder normalization (none or layer normalization, cf. Meyer, Pfaffl, and Ulbrich (2010)), w_{adve}, w_{reco}, w_{cyc}, and w_{GAN}, Figure 5 shows the results of five hyperparameter sets. Most sets were able to capture some distinctive features of each top clothing item but the relations with bottom clothing items are difficult to observe. There does not seem to be a heavy reliance on hyperparameters and we arbitrarily selected set 3 ($v \sim N(0.1I)$, no decoder normalization, $w_{adve}, w_{reco}, w_{cyc}, w_{GAN} = 10, 1, 1, 1$) for the remainder of this section.

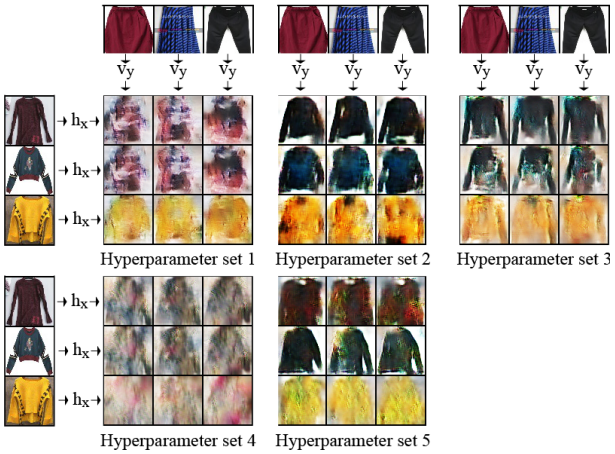

Figure 5: Comparison of five hyperparameter sets. Each generated image is produced by G_x from the corresponding v_y and h_x.

Next, our model is tested on known image-to-image translation datasets. These datasets are defined by that there is only one mapping rule that is human-interpretable. For example, in the 'edges2shoes' dataset (Zhu et al. 2017a) it would only make sense to pair a shoe image with an edge image if the contour of the shoe image is the same as the edge image.

In Figure 6, the results of application to the 'edges2shoes' dataset are shown. The network is first trained to minimize L^{ae}_{total} and L^{map}_{total}. Then, a single mapping function m_{deter} is learned from previous pairs. Figure 6 shows generated shoe images that have similar contours as the corresponding edge images although they do not match equally well as other state-of-the-art networks (Liu, Breuel, and Kautz 2017). This might indicate that the network is unable to encode enough information into vector v.

[1] tianchi.aliyun.com/competition/entrance/231670/information

Figure 6: Edges to shoes dataset. Mapping functions based on previous pairs.

Next, the network is tested on a novel task: generating top clothing to match bottom clothing. In this scenario multiple mapping rules are needed, because different people, in different situations, have different rules of how clothing should be paired. Here two rules are tested: (1) topology mapping, and (2) color matching. The goal is to evaluate our model's adaptivity to different rules. Adaptation to topology mapping is expected to generate similar items of top clothing when given similar images of bottom clothing. Adaptation to color matching is expected to generate top clothing that has the same color as the provided bottom clothing. Previous pairs for color matching are top and bottom items with similar color.

In Figure 7, topology mapping CDMN output is compared to that of UNIT (Liu, Breuel, and Kautz 2017) and color matching CDMN output is compared to that of Pix2pix (Isola et al. 2017). We see that while UNIT tends to generate tops that correspond to a given bottom (opposite color, similar texture, and shape), topology mapping CDMN finds a structural relation in which similar inputs result in similar outputs without correspondence of color, texture, and shape. Color matching CDMN can reproduce tops matching several major colors, although Pix2pix is more accurate in color matching. However, it is important to note that when the mapping rule changes, Pix2pix must learn from the beginning, while CDMN needs to find a new mapping function only.

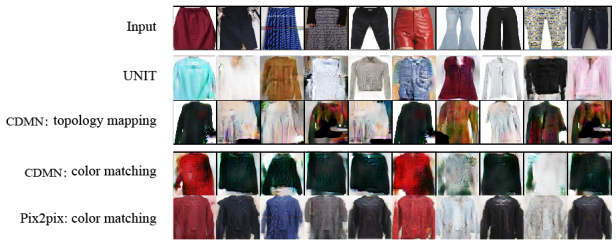

Figure 7: Comparison between outputs from different networks/ mapping functions.

Music visualization In this section, the model's performance on music-to-video mapping is assessed. Although we focus on video generation from music, but not vice versa,

since the network is trained bidirectionally even unidirectional generation partially illustrates the model's bidirectional behaviour.

The music dataset used is an arbitrary selection from the Million Song Dataset (Bertin-Mahieux et al. 2011) of 1000 songs that cover 10 genres and are reduced to 30 seconds duration each. The video dataset contains 10 videos of 200 frames each showing generated 'jumping' circles, as illustrated in Figure 8.

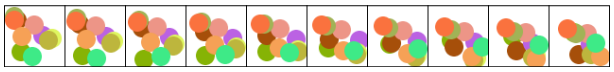

Figure 8: Ten successive frames of one video of the jumping circles dataset.

Figure 9 shows results from our model[2] with four hyperparameter sets, differing in music window sizes, cluster counts n_X, n_Y, and numbers of LSTM convolutional blocks in the decoder. Due to limited space we do not show the detailed configurations of the four sets, but they are not essential for later observations. We observe that Set 1 generates video that appears inconsistent and lacks an observable pattern corresponding to the music input signal. Set 2 generates a green blob in each frame that seems to expand with music signal amplitude increases. Set 5 appears to results in similar behaviour, but with very subtle frame differences. Using hyperparameter Set 7, the blue element at center-left of the frames appears to shrink in anticipation of rising music signal amplitudes, whereas it expands after the signal peak passes.

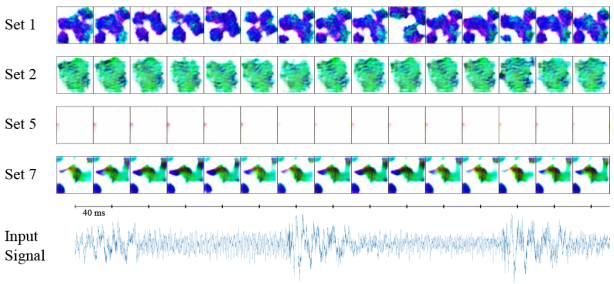

Figure 9: Illustration of video output for four different hyperparameter sets, and the music input signal. Each image in a row represents a frame with 40 ms interval (25 Hz).

By comparing these four hyperparameter sets, it becomes apparent that this model is difficult to tune for good video output; 'good' in the sense that viewers observe patterns in the video that temporally correspond with the music. This does not imply that the model cannot uncover patterns in the input music, but it could result in patterns within output videos that are too subtle to observe. Defining a proper criterion to ensure that changes in video are not too subtle nor too dramatic remains an open problem.

[2]Music visualization videos at vimeo.com/368386488

Creativity Evaluation

The next important question is: how creative is this model? We do not intend a thorough evaluation of creativity using external criteria in this paper. Instead we provide information and impression of the creative behaviour of our model from a technical point of view. We using the criteria mentioned earlier: (1) features are distributed and representative; (2) connections between domain spaces are flexible under different circumstances; and (3) flexible and persistent individuals can be modeled. The first criterion is inherently met through the application of autoencoders. We define two tasks to assess our model's agreement with criteria (2) and (3).

The first task focuses on the flexibility of mappings, specifically our model's behaviour in the face of environmental and intentional changes. Imagine a scenario where a fashion designer has her own style based on many years of experience. Recently she attended a fashion show, which subconsciously changed her style preferences. Then she met a new friend who is a famous fashion designer. Impressed by his talent she wants to mimic his style. Next, she found that her style was too similar to other designers, in particular to that of her friend, and she decides to create her own unique style. However, her old experiences and style are rooted and not easy to change.

Can our model mimic such human-like behaviour? To align this scenario with our artificial designer, let's assume that the designer's job is that, when provided with a clothing bottom (trousers or skirt), she must create a top. Her experience can be represented by the many pairs of tops and bottoms that she has seen (Guide Set 1, Figure 10), while the fashion show exposure is Guide Set 2, and her friend's style is Guide Set 3. Tops from all guide sets are randomly selected to match the given bottoms. Items in the set Creation 1 are generated solely based on Guide Set 1. Items in set Creation 2 are generated from Guide Sets 1 and 2 with random weights for all pairs of Set 1 in $[0.9, 1.1]$ and for Set 2 in $[0.8, 1]$. Items in set Creation 3 are generated from Guide Sets 1 and 2 with the same weights as previous and random weights for all pairs from Guide Set 3 in $[2, 3]$. Finally, set Creation 4 is generated with random weights for Set 1 in $[0.9, 1.1]$, for Set 2 in $[-0.2, 1]$, and for Set 3 in $[-3, -1]$. Topology mapping with a small weight is also added for each creation so clusters that are not covered by the guide sets can be mapped.

Results are shown in Figure 10. These include creations based on never-before-seen bottoms. We observe that Creation 1 matches Guide Set 1 well. Inclusion of Guide Set 2, as expected, only directly changes a few creations (e.g. columns 5 and 8). However, this small impact may also affect future creations. For example, in column 10 Guide Set 2 contains a pink-red top. After learning from Guide Set 3 which has no impact on column 10, the artificial designer creates a red top. Creation 3 matches with Guide Set 3 well. It is interesting to see that Creation 4 is similar but not identical to Creation 1. This implies that the artificial designer does not immediately disregard old experience (Guide Set 1) in the face of new experiences, but that its style is shifted slightly. The experiment shows that the

model can create flexible mapping functions under changing circumstances. These flexible behaviours simulate human creativity on some levels.

Figure 10: Guide sets and creations for flexibility evaluation. Each image of a top in a guide set is paired with the bottom in the same column. Each created image is based on the input bottom of the same column.

Our second task evaluates the modeling of persistent and flexible human individuals. While a persistent individual aims to find a single best solution, a flexible individual tends to explore for more possibilities, perhaps not all equally good. This difference can be modeled through the design choices of an evolutionary algorithm (EA). A persistent individual is modeled by a greedy EA with $(1 + \lambda)$ selection where λ are all offspring closest to the one parent. A flexible individual is modeled by an EA with $(15, 30)$ selection. Each (persistent or flexible) individual takes its previous individual as one solution in its first EA generation and then runs the EA to minimize the loss of topology mapping.

Here we do not have solid quantitative evaluation of music-visualization as it is very complex, if possible. Instead we evaluate it perceptually and qualitatively. We show that this already provides valuable information. Results are shown in Figure 11. We observe that persistent individuals 2 and 3 cannot create something new beyond persistent individual 1. Contrastingly, flexible individuals 2 and 3 are not restricted by prior experience encoded in flexible individual 1. However, judging by the loss, later generations of flexible individuals do not necessarily improve on earlier generations. Furthermore, even though persistent individuals have a lower loss, visually it is hard to say if the persistent individuals find 'better' mapping functions than the flexible individuals[3]. In fact, it is hard to assess whether or not an elaborated mapping function is better than a completely random mapping function!

Such difficulty implies that the model fails in carrying through relevant information from the music input to the visual output of the model. Perceptual consistency in the mapping appears missing. For music frames x_1, x_2, x_3, and mapping $x_1 \to y_1$, $x_2 \to y_2$ and $x_3 \to y_3$, if a listener finds that $\text{difference}(x_1,x_2) < \text{difference}(x_1,x_3)$, one would expect that visually y_1 appears more similar to y_2 than to y_3. This is however not perceived in the output of our model.

Besides potential optimization issues for the network, this also reveals the more profound problem that the nature of

[3]Visualization videos at vimeo.com/368390854

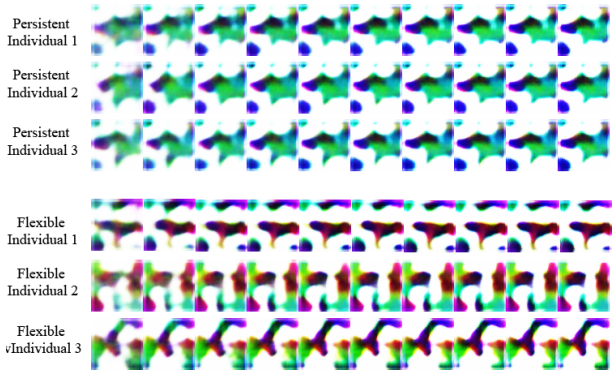

Figure 11: Six video sequences generated from the same music sample. Each row shows 10 successive frames, cf. Fig. 9.

finite cluster space and topology mapping make it hard to find a 'good' mapping. Clustering discretizes and originally infinite space, forcing areas of the original space to 'disappear' from consideration. One-dimensional topology mapping takes only Euclidean distance between clusters into account, implying that multi-dimensional relations between clusters are simplified and limiting the range of potentially generated output. In terms of the domains, would it make sense that a cluster of red jackets is closer to red T-shirts than to blue jackets? However, at this point, finite clustering with topology mapping is the only method for mapping. Future work may want to improve from here.

Conclusion

This work proposes the cross-domain mapping network (CDMN), a method for adaptive mappings and cross-domain content generation. It encodes finite cluster features and infinite individual variation thereupon from one domain, maps these cluster features to cluster features of a second domain, and from there generates (decodes) instances within the second domain. Different from previous work, the separation between encoding-decoding functions and mapping functions is modelled more towards replication of human creative behaviour and enables the mapping functions to adapt to changing situations. The use of mapping criteria based on topological distances within both domains and previous pairs helps the CDMN to show some complex human-like behaviours, as demonstrated in our scenario-based experiments.

We made an attempt to bridge creative cognition theory and machine learning applications. On one hand, as a GAN application this model achieves a higher level of creativity in terms of better adaptivity and individuality, when compared to prior work. Furthermore, as a realization of computational creativity theories, our model provides a highly automated method with which the unitary action control model with feature distribution and connection is shown to be computable. It shows the possibilities of using machine learning tools as a convenient and powerful method to build creative models and evaluate theories about creative cognition and

psychology.

Possible future work is suggested in two directions. In the direction of computational modelling, the limitations brought by finite clustering and topology mapping should be addressed. It is also possible to construct continuous feature spaces and design mapping functions conditional on regionality within those feature spaces, as opposed to applying indexed clustering in feature spaces. Moreover, it is not trivial to tune hyperparameters as more in-detail analysis can be performed with more detailed models. In the direction of cognition theories, mapping rules used in this paper can arguably model somewhat but limited human-like behaviors. This is because the mapping rules proposed in this paper are ad-hoc, which is due to the fact that how mapping functions are controlled is not well-known in cognitive psychology (Hommel and Wiers 2017). This paper addresses the importance of such studies to achieve closer-to-human level creativity in computational models.

References

Augello, A.; Infantino, I.; Lieto, A.; Pilato, G.; Rizzo, R.; and Vella, F. 2016. Artwork creation by a cognitive architecture integrating computational creativity and dual process approaches. *Biologically Inspired Cognitive Architectures* 15:74–86.

Bertin-Mahieux, T.; Ellis, D. P.; Whitman, B.; and Lamere, P. 2011. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*.

Ephrat, A., and Peleg, S. 2017. Vid2speech: speech reconstruction from silent video. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2017)*, 5095–5099. IEEE.

Gabora, L. 2010. Revenge of the "neurds": Characterizing creative thought in terms of the structure and dynamics of memory. *Creativity Research Journal* 22(1):1–13.

Guilford, J. P. 1967. Creativity: Yesterday, today and tomorrow. *The Journal of Creative Behavior* 1(1):3–14.

Hinton, G. E., and Salakhutdinov, R. R. 2006. Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507.

Hommel, B., and Wiers, R. W. 2017. Towards a unitary approach to human action control. *Trends in cognitive sciences* 21(12):940–949.

Huang, X.; Liu, M.-Y.; Belongie, S.; and Kautz, J. 2018. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV 2018)*, 172–189.

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*, 1125–1134.

Kajić, I.; Gosmann, J.; Stewart, T. C.; Wennekers, T.; and Eliasmith, C. 2017. A spiking neuron model of word associations for the remote associates test. *Frontiers in Psychology* 8:99.

Kenett, Y. N.; Levy, O.; Kenett, D. Y.; Stanley, H. E.; Faust, M.; and Havlin, S. 2018. Flexibility of thought in high creative individuals represented by percolation analysis. *Proceedings of the National Academy of Sciences* 115(5):867–872.

Kingma, D. P., and Welling, M. 2014. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR 2013)*.

Liu, M.-Y.; Breuel, T.; and Kautz, J. 2017. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, 700–708.

Makhzani, A.; Shlens, J.; Jaitly, N.; Goodfellow, I.; and Frey, B. 2015. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*.

Mednick, S. 1962. The associative basis of the creative process. *Psychological Review* 69(3):220.

Mekern, V.; Hommel, B.; and Sjoerds, Z. 2019. Computational models of creativity: a review of single-process and multi-process recent approaches to demystify creative cognition. *Current Opinion in Behavioral Sciences* 27:47–54.

Meyer, S. U.; Pfaffl, M. W.; and Ulbrich, S. E. 2010. Normalization strategies for microRNA profiling experiments: a 'normal' way to a hidden layer of complexity? *Biotechnology Letters* 32(12):1777–1788.

Olteţeanu, A.-M., and Falomir, Z. 2016. Object replacement and object composition in a creative cognitive system. Towards a computational solver of the Alternative Uses Test. *Cognitive Systems Research* 39(C):15–32.

Song, Y.; Zhu, J.; Li, D.; Wang, A.; and Qi, H. 2019. Talking face generation by conditional recurrent adversarial network. In *Proceedings of the 28 International Joint Conference on Artificial Intelligence, IJCAI 2019*, 919–925.

Taigman, Y.; Polyak, A.; and Wolf, L. 2017. Unsupervised cross-domain image generation. In *Proceedings of the 5th International Conference on Learning Representations (ICLR 2017)*.

Xingjian, S.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.-K.; and Woo, W.-c. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, 802–810.

Yoo, D.; Kim, N.; Park, S.; Paek, A. S.; and Kweon, I. S. 2016. Pixel-level domain transfer. In *Proceedings of the European Conference on Computer Vision (ECCV 2016)*, 517–532. Springer.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017a. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2017)*, 2223–2232.

Zhu, J.-Y.; Zhang, R.; Pathak, D.; Darrell, T.; Efros, A. A.; Wang, O.; and Shechtman, E. 2017b. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems*, 465–476.