Co-Creative Drawing Agent with Object Recognition

Nicholas Davis, Chih-Pin Hsiao, Kunwar Yashraj Singh, Brian Magerko
Georgia Institute of Technology
School of Interactive Computing
{ndavis35, chsiao9, kysingh, magerko}@gatech.edu

Abstract

This paper describes an updated version of a co-creative drawing system called the Drawing Apprentice. The system collaborates with users by analyzing their drawn input and responding in a real time dialogical and improvisational interaction. The current system includes an object recognition module that employs deep learning to classify sketched objects. The system architecture and implementation are described along with its evaluation during a public demonstration during which artists, non-artists, and designers provided feedback about the experience interacting with the system.

Introduction

Collaboration has the potential to inspire creativity in novel ways as each individual interprets and builds upon ideas through time in unexpected ways (Mamykina, Candy, & Edmonds, 2002; Sawyer, 2000; Uzzi & Spiro, 2005). Collaboration yields a new type of *distributed creativity* wherein products and ideas emerge through improvised interaction (Sawyer & DeZutter, 2009). Each individual has a unique lens and viewpoint based on their own experience through which they may interpret and add to the creative product (Mamykina et al., 2002). As a result, each participant's contribution can change the course of the creative product and suggest new ideas and themes to explore further, resulting in a potentially more creatively engaging and fulfilling experience.

Recently, the field of computational creativity began exploring how creative agents might engage in collaboration with humans. These *co-creative agents* directly collaborate with users on creative tasks as an equal partner or colleague in the creative process by making independent contributions to a shared creative product (Davis, 2013; Yannakakis, Liapis, & Alexopoulos, 2014). These types of systems are a hybrid between creativity support tools that are meant to help users accomplish creative tasks (Edmonds & Candy, 2005; Shneiderman et al., 2006) and generative systems that produce creative products autonomously (Colton, Goodwin, & Veale, 2012; Misztal & Indurkhya, 2014; Norton, Heath, & Ventura, 2014). Co-creative agents, also referred to as *computer colleagues* (Lubart, 2005), offer opportunities to push forward the field of computational creativity as well as powerful new ways to interface with and augment the creative process through collaboration.

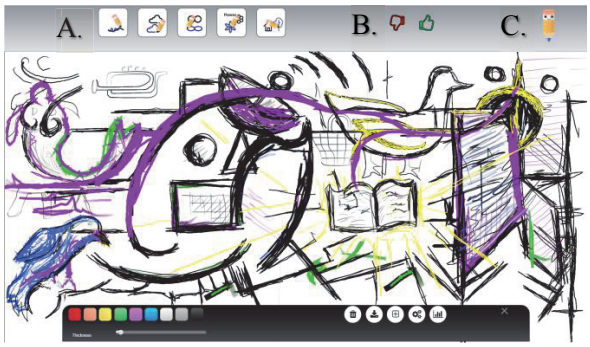

Figure 1. The Drawing Apprentice Interface and Example Drawing. Top panel offers the communication channel between the user and agent. Bottom panel contains conventional drawing functions.

The *Drawing Apprentice* was developed as a technical probe to explore the potential for co-creative colleagues to inspire creativity in open-ended improvisational domains like collaborative drawing. The Drawing Apprentice is a co-creative drawing agent that analyzes the user's input and responds with artistic contributions of its own on a shared canvas. The system is designed to help inspire new ideas and facilitate *distributed creativity* by co-creating shared meaning in the collaboration.

The collaborative drawing in Figure 1 was done by the system and the first author who has significant experience with abstract drawing. The artist began the drawing in object recognition mode because he wanted the system to provide some representational ideas that he could build upon. During this phase, turn taking was clearly delineated since the

artist waited to see what the system did before beginning his next turn. All of the representational objects in the drawing were initially generated by the system and later refined and added upon by the artist. The book in the center emerged as a unifying theme and the artist began to visually connect it to other regions. In this phase, the artist used the mimicry and transformation drawing modes to help create a variety of lines and additional ideas for connecting the different regions. Finally, the artist began refining and thickening lines and adding color to finish the piece. He used the tracing drawing mode to get the system to help him thicken the lines. During refinement, collaboration became more synchronous as the artist did not wait to see how the agent responded. He simply assumed the agent would be performing similar activities as him and periodically checked what the agent had done. In some cases, the artist would stop his current refinements to add upon or correct lines drawn by the agent.

Previous versions of the system (Davis et al., 2014, 2015; Davis, Hsiao, Yashraj Singh, Li, & Magerko, 2016) focused on imitating the user's sketch input using various sketch transformation algorithms. In these previous prototypes, user feedback was employed to train the system about the user's stylistic preferences. Previous publications have reported the results of the experiments with a mixed evaluation method on these versions of the Drawing Apprentice (Davis et al., 2016).

In this paper, we emphasize the ways of understanding the user's drawing and responding with appropriate objects by employing object recognition. Implementing object recognition in a real time and open-ended creative context presents significant challenges both in terms of how to structure the sketch input data for classification as well as ensuring a high accuracy in a short amount of time to facilitate real time interaction. Thus, the primary contributions of this paper are describing our implementation of real-time sketch-based object recognition using deep learning and its reception in a collaborative drawing context with different user groups. Approximately 20 individuals (including artists, non-artists, and designers) used this new version of the system during public demonstrations and provided informal feedback about their experiences, which revealed many productive applications for each group.

Related Work

The work reported here is an interdisciplinary effort nestled between the fields of computational creativity and creativity support tools. Computational creativity is a field of artificial intelligence focused on developing agents that generate creative products autonomously (Boden, 1990; Colton, Wiggins, & others, 2012; Wiggins, 2006). Creativity support tools, on the other hand, are technologies designed to enhance and augment the user's creativity, typically aiming to improve the quality of the final product (Edmonds & Candy, 2005; Schneiderman, 2007; Shneiderman et al., 2006; Voigt, Niehaves, & Becker, 2012). Computers can support human users in their creative process in a variety of ways, including acting as a nanny, coach, pen-pal, and colleague (Lubart, 2005) . The Drawing Apprentice can be considered a computer colleague since it collaborates with users on their creative task.

Tagawa & Unemi present a tool called Co-Drawing System that draws with users on a doodling task (Tagawa & Unemi, 2014). The system aims to produce a feeling of empathy between the human and user in order to facilitate a fun and engaging drawing session. Two factors found to be important for the system's drawing style were the speed with which lines were drawn on the canvas and the location (i.e. distance from user input). When these factors deviated too far from user input, the agent was evaluated as a less effective collaborative partner.

Previous work investigating collaborating with a co-creative drawing agent found that spatial awareness, visual similarity, and perceptual logic were important factors that contributed to whether the agent's contributions 'made sense' to the user (Davis et al., 2016). Spatial awareness refers to the agent respecting existing shapes, boundaries, and drawing in a similar region as the user. Visual similarity describes how users could more easily understand contributions that were clearly visually related to their own. Finally, users expected the system to understand what they were drawing, including the 'logic' behind the application of patterns they were producing and objects they were drawing. Participants reported wanting the system to contribute to their drawing in a way that reflected some understanding of what they were drawing. For this reason, the authors concluded object recognition is an important skill for a co-creative drawing agent. Enabling the agent to understand what types of objects are being drawn opens up the opportunity for more coordinated collaboration through shared understanding.

The object recognition approach used in the Drawing Apprentice system extends previous work by Eitz et al (Eitz, Hays, & Alexa, 2012). These authors collected a large corpus of human-drawn objects that included 250 unique categories (e.g. airplane, sun, flower) with 80 different instances of each object for a total of 20,000 examples. Eitz et al. used an SVM machine learning technique to achieve a recognition accuracy of approximately 55% (Eitz, Hays & Alexa 2012). More recently, Yu et al. (2015) applied a deep learning approach to recognizing objects from this same dataset achieving a recognition rate of approximately 74.9%, which beat human recognition rates of 73.4% (Yu et al. 2015).

For our use-case, neural network approaches are preferred over SVM approaches since these networks learn features directly from the data rather than from human input, which enables learning of new objects more efficiently. One significant risk for neural network approaches, however, is overfitting to the training data such that novel inputs are recognized with less accuracy. This issue is particularly relevant for our target application of collaborative drawing

given the wide variety of styles and representations of sketched objects employed by users. To circumvent this overfitting issue, we employed a variety of data augmentation methods explained in more detail in the sketch classification section.

System Overview

In the domain of drawing, there are a number of critical variables for a co-creative agent to consider when devising its sketch contribution, such as *what* to draw, *when* it should be drawn, *where* to place it on the canvas, *how* it should be drawn (i.e. the manner of drawing the lines), and *why* the agent should draw a particular element. Simultaneously determining what the 'correct' answer is for these variables in an open-ended collaboration presents a monumental challenge for a creative agent given the nearly infinite variety and dynamism of the artistic intentions of users throughout their creative process. To delimit the creative responsibilities of the agent and help facilitate a coordinated collaboration, our approach offloads some of the creative decision making processes onto the user through direct interface controls and feedback mechanisms. This hybrid approach enables the user to maintain a degree of control over the agent's drawing activities while still affording creative and unexpected contributions within defined boundaries.

To explain the functionality of the system, we will begin by describing the user interface and its features. The interface has three main components: a palette of functions to control the agent, a palette of drawing tools, and a shared canvas. The agent palette contains buttons for controlling the five drawing modes (section A of Figure 1), voting buttons for providing feedback (section B of Figure 1), and the 'home base' of the character icon representing the co-creative agent (section C of Figure 1). The agent's interpretations of the user's drawing activities appear as a speech bubble in this home-base. The drawing palette consists of functions traditionally associated with drawing applications, such as selecting a color, line thickness, saving the image, and starting a new canvas.

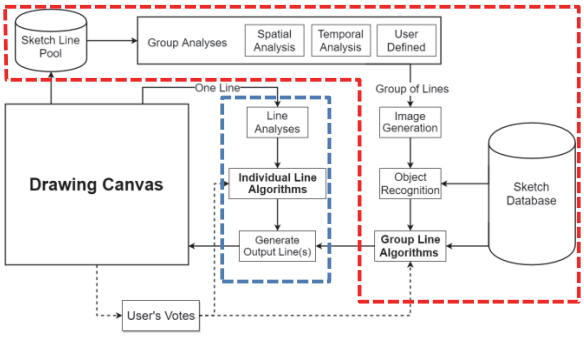

Figure 2. System Diagram. We introduced the parts in the blue dashed box in the previous paper (Davis et al. 2016). In this paper, we focus on the red parts.

After the agent finishes its turn, the user can provide feedback via voting buttons to inform the agent whether the user liked its contribution. This voting information is used to learn the aesthetic preferences of each user and fine tune what types of contributions it would make by using a Q-Learning algorithm described in our previous paper (Davis et al., 2016). When a drawing mode is selected, the image on the button animates to provide a prototypical demonstration of what the drawing mode entails to help the user understand what to expect from the system.

The drawing modes can be divided into two functional categories: responding to individual lines (the blue block in Figure 2), and responding to groups of lines (red block in Figure 2). The first three drawing modes all respond to individual lines by either 1) *tracing* the user input, 2) *transforming* user input (e.g. scaling, rotating, translating), or 3) *mimicking* user input by stretching and skewing it (Davis et al., 2016). This category of drawing modes was designed primarily for use in abstract drawing to provide users with novel input to stimulate ideas. However, findings from user studies (Davis et al., 2016) indicated the needs of responding representational objects with a clear artistic intention.

To enable the system to collaborate with representational contributions, we created the second category of drawing modes that groups input lines and attempts to classify the type of object the user is drawing, as illustrated in the red block of Figure 2. First, the system must determine which lines to group (see Line Grouping section), then an image is formed from those lines and sent to a precomputed convolutional neural network model for classification. Finally, the system employs one of the grouped-line drawing algorithms and outputs the results to the shared canvas.

In the following sections, we describe the means for responding representational objects, including grouping sketch input, classifying sketch input, selecting what to draw, determining where to draw on the canvas, and finally animating the drawing contribution on the canvas in an embodied manner.

Line Grouping

One of the significant challenges for implementing object recognition in an open-ended drawing application is determining which lines to group together to send to the sketch recognition module. One solution would be to offload this task onto the user by having them manually group lines. While this solution is potentially the most accurate, forcing users to manually group every object would significantly disrupt their creative flow, which is an important design consideration for the present application. Automatic and implicit grouping is therefore greatly preferred, but this issue is complex because individuals may begin an object and return to it later, meaning that the input is separated in time but co-located in space. To address this challenge, we developed a three tiered solution for grouping sketched lines: (1) time-based implicit grouping, (2) space-based implicit

grouping, and (3) explicitly assigned grouping through user input. While the last method needs user input, the first two methods occur without any intervention from users.

In the *time-based implicit grouping* method, the system starts a timer every time the user lifts their pen from the sketch canvas. If a pre-specified period of time passes between strokes, the system assumes the user has completed a full 'turn' to fully express their idea, and it will mark the last stroke as an "end stroke." Based on our observation, we set this interval to 3 seconds. After the time is up, it groups all of the strokes between the previous "end stroke" and the current "end stroke" as one turn. These strokes are rendered as a small temporary image isolated from the other strokes on the canvas, and fed into the sketch classification procedure to classify the sketch.

In the *space-based implicit grouping method*, the system constructs a quadtree data structure that includes all the points from strokes collected from the human users and AI agent. In this quadtree, the data in the regions that have a higher density of lines will be contained in the nodes with higher depths. Once one particular node is four levels deeper than the average depth of the tree, it returns the area surrounding the node as an 'area of interest.' Then, the system draws an image from the selected strokes in this area similar to the time-based method for sketch classifications. This approach helps to reduce the computational power required for the common computer vision analyses and helps ensure real-time responses.

Users can also manually group sketches in the canvas using a lasso tool in the UI for sketch classification. The user can choose to manually label the object themselves to serve as another 'ground truth' example to help improve the sketch recognition model.

Sketch Classification

We employed convolutional neural networks to classify sketch input due to their recent breakthrough in image and sketch recognition even though methods like Bag-of-Words and SVM worked well in past (Eitz, Hays & Alexa 2012; Yu et al. 2015). Since convolutional neural networks also have cognitive and psychological plausibility given that they operate directly on images, similar to the visual cortex in the human brain, we decided to have an end-to-end learning mechanism instead of going the feature engineering route like Bag-of-Words.

Our sketch classification model was inspired by VGG neural network due to its recent success in large-scale image recognition. We modified the VGG-CNNs and VGG-19 architecture to suit our task. Since both of these deep neural network models deal with images that contain texture information (encoded using R,G,B channels), we reduced the number of channels to just one as sketches can be represented as binary images (Chatfield, Simonyan, Vedaldi, & Zisserman, 2014). Furthermore, we removed the Local Response Normalization layers from these networks as we found that they work well with images that contain textural information but not well with the task of recognizing sketches (Yu et al. 2015). To reduce overfitting, we made use of data augmentation where we randomly flipped horizontally and scaled the training images in addition to using a higher dropout rate of 50% in the last two fully connected layers.

The requirement for having a real time sketch classification engine favored the VGG CNNS model as it has less parameters (and resultantly takes less time to feedforward) than VGG-19 (Simonyan & Zisserman, 2014), even though the VGG-19 model provided a greater classification accuracy. Therefore, we decided to move with the VGG CNN-S architecture.

VGG models have stacks of convolution layers with smaller filter sizes compared to the Sketch-a-Net architecture with large filter size and high strides (Yu et al. 2015; Simonyan & Zisserman 2015). Smaller filter size helps detect local sketching patterns such as crosshatches in conjunction to the overall sketch. The main difference between Sketch-a-Net and our model is that Sketch-a-Net uses a multi-channel and multi-scale pipeline with stroke ordered training data whereas, our model operates on a single scale and single channel and the strokes in the training data are not ordered. As a result, our current classifier is given a very limited set of information about the sketch. The training of the network was done on the TU-Berlin sketch database, which has 250 categories with each category having 80 different example sketches (Eitz, Hays & Alexa 2012). During the training phase we split 90% of the data into the training set and the remaining 10% of the data as test set.

HOG-SVM	Sketch-a-Net	Le-Net	VGG-CNN-S (modified)
56%	74.99%	55.2%	63.9%

Table 1: Comparison of classification accuracy for different state of the art methods

Through our experiments we found that other optimization algorithms such as ADAM, AdaMax and RMSProp did not work that well for training these models (Kingma & Ba, 2014). Furthermore, we found through experimentation that the learning rate should be as low as possible, which was 0.001, in order to train the network incrementally. Hence, we made use of Stochastic Gradient Descent with Nesterov Momentum with a learning rate of 0.001 and momentum of 0.9 to train these models. Keeping the learning rate to a minimum helped to counter overfitting to the training data and helped reach an accuracy of 63.95% in 100 epochs. VGG-CNN-S was trained on only one image size of 224 by 224 and tested using the hold out testing method. The interesting thing we noted was that the VGG-CNN-S trained and reached decent accuracy in less number of iterations, making it a suitable candidate that can be used in a real-time sketch learning environment. Table 1 compares the accuracy of popular deep neural networks on sketch data.

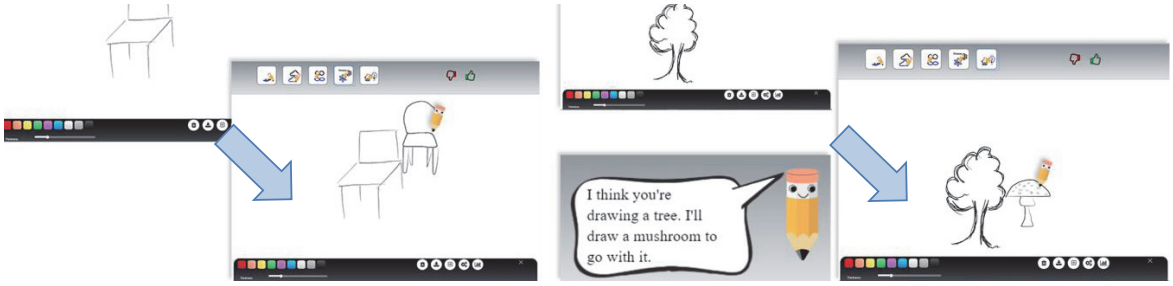

Figure 4: Drawing modes using sketch recognition to draw similar (left) and complimentary (right) objects next to the user's most recently drawn object. The agent explicitly expresses what it recognizes and plans to draw (middle).

Object Placement

One of the primary findings of previous user studies pointed to spatial awareness as one of the primary needs for a co-creative drawing agent (Davis et al., 2016). In this project, we have two main criteria when finding a location for the agent to draw a new object: (1) empty region where the targeting drawing object would minimally intercept with existing objects; and (2) close to the object that are drawn in recent turns. Here, we employ the same quadtree data structure mentioned before for further analyses that is utilized in determining object placement in real-time.

Once the system detects a turn, it uses the bounding rectangle formed by the users' sketches to find an area to draw. As shown in Figure 3, the system iterates through the surrounding locations starting from the top-left corner of to the sketch from the user's current turn first, and then queries the quadtree to get a candidate bounding rectangle containing the least packet points for drawing area. This approach ensures the target object is drawn as close to the user's previous input as possible without drawing on top of existing elements. Figure 4 shows examples of the user's sketch and the locations where the system picks for drawing. With the results of turn detection, sketch classification and placement, the system utilizes the following two modes for generating the new sketch objects.

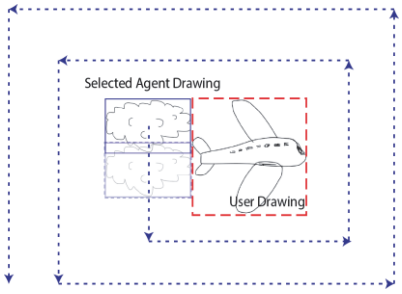

Figure 3. Iterative spatial search procedure to find target drawing area near user's object.

Drawing Similar Objects Mode

In this drawing mode, the system recognizes the user's drawn object and then responds with a different representation of that same object. Figure 4-left shows an example where the user drew a chair in the perspective view. The system responded with another chair similar to the original chair. The system uses the t-SNE algorithm on the visual features extracted by the convolutional neural network to compute the nearest neighbor image in 2-dimensional embedding of the features. This method provides the ability to draw visually similar or dissimilar objects (relative to the user input) of the target category.

Drawing Complimentary Objects Mode

In this mode, rather than drawing an object from the same classification, the system selects a semantically related category and then randomly picks an object from that category. The right side of Figure 4 shows that the system recognized a tree has been drawn by the user, then responded with a message in a speech bubble stating its interpretation and planned contribution, and finally drew a mushroom on the canvas. To pick a category, we manually created a dictionary that categorizes the sample sketches into 15 high-level categories (with several sub-categories) based on their semantic meanings. For instance, we group all the animals as one category with marine, bird, and land animals as subcategories.

Ideally, the system should utilize existing concept nets, such as ConceptNet3 (Havasi, Speer, & Alonso, 2007) as well as learn new relationships by observing what objects users typically draw together. As shown by the literature on concepts and categories, these elements are subject to change based on context and intention (Lakoff, 1999). To account for this plasticity, we plan to implement a module that analyzes which objects tend to be drawn together and use this data to inform this algorithm in the future.

System Evaluation in the Wild

The Drawing Apprentice system was exhibited during a public event at which many interactive technologies were demonstrated to individuals from the local community. Over the course of 3 hours, approximately 20 people observed and interacted with the system and provided informal verbal feedback about their impressions. Self-reported artistic experience was noted for inclusion in a thematic analysis that revealed three distinct user groups: practicing artists, visual designers, and non-artists. A clear delineation of use cases and needs emerged among these three user groups that will be expanded upon in the following sections. This type of informal feedback 'in the wild' is helpful in understanding what role co-creative agents may play in the creative process of different users.

Non-Artists

Non-artists reported enjoying instances where the system responded to their input with complimentary objects, i.e. users draw an eye and the system draws eyeglasses. The dialogical turn-taking component of the interaction prompted them to respond and continue the interaction, which may help prevent task-abandonment. Instead of worrying about the final outcome, novices would be more focused on responding to their partner with an interesting and creative contribution that builds on what has been previously contributed. By building on the user's contribution and offering new ideas to explore, the Drawing Apprentice has the potential to engage non-artists in a *creative conversation* through drawing that inspires their creativity in a way that could be both entertaining and cognitively stimulating.

Artists

Unlike non-artists, artists were strongly concerned with how the system might help them draw better and 'get things done.' While they agreed it could help with creative inspiration, artists wanted the system to attempt to understand and predict their 'creative trajectory,' i.e. where they are headed in terms of the artwork based on what has been done thus far. Importantly, artists want to have a means of viewing and manipulating the creative trajectory the system calculates to increase their control over the agent's activities.

Designers

Designers that engaged with the system focused on how it might help with ideation and *pair brainstorming*. During pair brainstorming, two individuals engage in a collaborative design session where they each come up with different versions of a target design. This type of brainstorming helps designers fully explore the design space and help understand the design problems. They noted that the Drawing Apprentice could perform the role of their partner so they may engage in this productive form of collaborative brainstorming more often without a human partner. In particular, these designers liked how the system would mimic their designs with slight alterations in unexpected ways, or drew different versions of the same object.

Future Work & Conclusions

The immediate next steps for the project focuses on children as a user group, specifically studying whether this type of co-creative agent can help encourage kids to express their ideas and explore more ideas throughout a drawing. Just as a parent might add upon a child's drawing to help inspire their creativity, the Drawing Apprentice could recognize what objects children are drawing and add thematically related objects to help sustain the child's engagement on the drawing task and inspire new ideas.

The technical requirements of the child-focused use case revolve around enabling the system to reason about the semantic and spatial relations between objects (in addition to recognizing individual objects). Further, there is spatio-temporal data encoded in the construction of scenes, such as the order in which the objects are typically drawn, the scale of each object (i.e. flowers should be smaller than trees), and the positioning of each object relative to each other and the overall composition. We refer to this technical component as a 'narrative module' since it deals specifically with which objects should appear in a sequence together, like parts of a narrative. This narrative learning module would be well suited for learning 'themes' or 'scenes' that occur often in drawings, such as a house scene that might include a house, tree, person, animal, sun, and clouds.

This paper described the implementation and informal evaluation of a co-creative drawing agent with object recognition. We identified challenges for implementing real-time object recognition in the open-ended context of collaborative drawing, and methods for overcoming these challenges. We presented insights from an informal evaluation that helped identify the creative needs and perspectives of diverse user groups, including artists, non-artists, and designers. This work adds to the growing literature exploring how co-creative agents might collaborate with users in open-ended improvisational creative domains.

Acknowledgements

This work was supported by NSF grant IIS #1641008.

References

Boden, M. A. (1990). *The Creative Mind: Myths and Mechanisms*. London: Weidenfeld & Nicolson.

Chatfield, K., Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. *arXiv Preprint arXiv:1405.3531*.

Colton, S., Goodwin, J., & Veale, T. (2012). Full face poetry generation. In *Proceedings of the Third International Conference on Computational Creativity* (pp. 95–102).

Colton, S., Wiggins, G. A., & others. (2012). Computational creativity: The final frontier? In *ECAI* (pp. 21–26).

Davis, N. (2013). Human-Computer Co-Creativity: Blending Human and Computational Creativity. In *Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Davis, N., Hsiao, C.-P., Singh, K. Y., Li, L., Moningi, S., & Magerko, B. (2015). Drawing Apprentice: An Enactive Co-Creative Agent for Artistic Collaboration. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition* (pp. 185–186).

Davis, N., Hsiao, C.-Pi., Yashraj Singh, K., Li, L., & Magerko, B. (2016). Empirically Studying Participatory Sense-Making in Abstract Drawing with a Co-Creative Cognitive Agent. In *Proceedings of the 21st International Conference on Intelligent User Interfaces* (pp. 196–207).

Davis, N., Popova, Y., Sysoev, I., Hsiao, C.-P., Zhang, D., & Magerko, B. (2014). Building Artistic Computer Colleagues with an Enactive Model of Creativity. In *International Conference on Computational Creativity*. Ljubljana, Slovenia: AAAI.

Edmonds, E. a., & Candy, L. (2005). Computer support for creativity. *International Journal of Human-Computer Studies*, *63*(4-5), 363–364. http://doi.org/10.1016/j.ijhcs.2005.04.001

Eitz, M., Hays, J., & Alexa, M. (2012). How do humans sketch objects? *ACM Transactions on Graphics*, *31*(4), 1–10. http://doi.org/10.1145/2185520.2335395

Havasi, C., Speer, R., & Alonso, J. (2007). ConceptNet 3: a flexible, multilingual semantic network for common sense knowledge. In *Recent advances in natural language processing* (pp. 27–29).

Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv Preprint arXiv:1412.6980*.

Lakoff, G. (1999). Cognitive models and prototype theory. *Concepts: Core Readings*, 391–421.

Lubart, T. (2005). How can computers be partners in the creative process: Classification and commentary on the Special Issue. *International Journal of Human-Computer Studies*, *63*(4-5), 365–369. http://doi.org/10.1016/j.ijhcs.2005.04.002

Mamykina, L., Candy, L., & Edmonds, E. (2002). Collaborative creativity. *Communications of the ACM*, *45*(10). http://doi.org/10.1145/570907.570940

Misztal, J., & Indurkhya, B. (2014). Poetry generation system with an emotional personality. In *Proceedings of 5th International Conference on Computational Creativity, ICCC*.

Norton, D., Heath, D., & Ventura, D. (2014). Autonomously managing competing objectives to improve the creation and curation of artifacts.

Sawyer, R. K. (2000). Improvisational Cultures: Collaborative Emergence and Creativity in Improvisation. *Mind, Culture, and Activity*, *7*(3), 180–185. http://doi.org/10.1207/S15327884MCA0703_05

Sawyer, R. K., & DeZutter, S. (2009). Distributed creativity: How collective creations emerge from collaboration. *Psychology of Aesthetics, Creativity, and the Arts*, *3*(2), 81.

Schneiderman, B. (2007). Accelerating Discovery and Innovation. *Communications of the ACM*, *50*(12).

Shneiderman, B., Fischer, G., Czerwinski, M., Resnick, M., Myers, B., Candy, L., … Terry, M. (2006). Creativity Support Tools: Report From a U.S. National Science Foundation Sponsored Workshop. *International Journal of Human-Computer Interaction*, *20*(2), 61–77. http://doi.org/10.1207/s15327590ijhc2002_1

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv Preprint arXiv:1409.1556*.

Tagawa, S., & Unemi, T. (2014). On Effects of Cooperation with the Machine in Human-Computer Co-Drawing. In *XVII Generative Art Conference* (pp. 307–315).

Uzzi, B., & Spiro, J. (2005). Collaboration and creativity: The small world Problem1. *American Journal of Sociology*, *111*(2), 447–504.

Voigt, M., Niehaves, B., & Becker, J. (2012). Towards a Unified Design Theory for Creativity Support Systems, (GroupSystems 2011), 152–173.

Wiggins, G. A. (2006). Searching for computational creativity. *New Generation Computing*, *24*(3), 209–222.

Yannakakis, G. N., Liapis, A., & Alexopoulos, C. (2014). Mixed-initiative co-creativity. In *Proceedings of the 9th Conference on the Foundations of Digital Games*.

Yu, Q., Yang, Y., Song, Y.-Z., Xiang, T., & Hospedales, T. M. (2015). Sketch-a-net that beats humans. In *Proceedings of the British Machine Vision Conference (BMVC)* (pp. 1–7).