3D Visualization of a Constrained Hidden Markov Process

Porter Glines[*]
Department of Computer Science
Idaho State University

ABSTRACT

There is interest in making generative models more explainable and in increasing the intentionality of systems. Visualization can be used to provide insights into the inner workings of a model. The focus of this paper is on a constrained hidden Markov process. While a limited visualization exists for a simpler constrained non-hidden Markov model, the visualization in this paper aims to interactively present the entire, more complex, constrained hidden Markov model.

To do so, a three dimensional visualization is created using the WebGL library: ThreeJS. A graph-based visual is created using Tufte's guidelines focusing on maximizing data-ink ratios and aesthetic elegance [8].

Source code is available at *https://github.com/po-gl/ChimpVisual*

1 INTRODUCTION

Sequence generation is a common task to Artificial Intelligence (AI) or Computationally Creative (CC) systems. Sequence generation models have been applied to domains such as natural language [10] [7], music [9], etc; one such sequence generation model is the constrained hidden Markov model [5].

In CC systems, the concept of *intentionality* is an important characteristic to how effective or creative a system is. Dan Ventura provides a definition in his paper ”How to Build a CC System” [15]:

> **intentionality:** the fact of being deliberative or purposive; that is, the output of the system is the result of the system having a goal or objective—the system's product is correlated with its process. [15]

A closely related concept to intentionality is that of explainable CC systems [6], which is defined below:

> **explainability:** The ability of a system to show how it came to its conclusion.

The goal of this paper is to increase the intentionality and explainability through three dimensional visualization of systems using the constrained Hidden Markov process. The visualization exposes the inner workings and connections of the model through a graph-based visual that shows nodes and edges representing structural characteristics of the model. Artifact characteristics can be seen in the visual such as words and parts-of-speech in satisfactory sequences. To address the complexity and expanse of the model, the visual is interact-able in a three dimensional environment that allows exploration through camera orbit, pan, and zoom interactions. Furthermore, model complexity is addressed through selectively hiding parts of the model—only showing one selective observed node ”word cloud” per sequence position.

Through visualization, model intentionality and model explainability is increased by showing viewers the inner workings and connections of the model which leads to the sense that the model is doing more than picking words at random and leads to the model being able to show how it generated an artifact. The remainder of the paper is structured as follows: background and related works are provided, details regarding the methods involved in creating the visual and results are shown, a discussion on the visual is provided, and finally future work is described and the paper is concluded.

2 BACKGROUND AND RELATED WORKS

Computational creativity is a sub-field to the more broad field of artificial intelligence (AI). It is interesting to note the parallel struggle that both CC and AI face in defining inherently subjective terms. CC struggles to define what creativity is while AI struggles to define what intelligence is. A satisfying definition of the field has yet to be found; however, Colton and Wiggins [2] provide a definition that has been generally accepted in the past few years as a good enough stand-in for a proper definition:

> *[Computational creativity is] the philosophy, science and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviours that unbiased observers would deem to be creative.*

As mentioned in the introduction, intentionality is a characteristic of a CC system which lends itself to exhibiting behaviors that an observer would deem to be creative. A system with more intention behind its actions is typically deemed as exhibiting more creative behavior [15] [14] [4]. There is a story illustrating the importance of intentionality/explainability within the CC community that goes something like the following:

> A person is attending an art exhibition in which they find themselves standing in front of two very similar paintings. Both artists are present and discuss their paintings with this person. The first artist explains that the painting was merely an accidental splattering of paints. But the second artist explains their inspiration for each section of their composition—where each inspiration is meaningful. The person comes back to the art exhibition the next day with the intention to buy the painting from the second artist. However, neither artist is attending that day and they cannot tell which painting came from the second artist. Because they cannot tell which painting is merely random and which was inspired, they decide to buy neither painting.

The story illustrates how an artifact of a creative system can become worthless without intentionality or explainability behind its inception. Because of this, conveying to a user how the system is intentional is important to the perceived value of the system's output. In the context of natural language, conveying the intricacies of the underlying model would be a way to show a user that the system is not merely choosing random words.

Hidden Markov processes generate sequences by looking at the previous state and randomly sampling according to trained hidden and observed probabilities to generate a new token—non-hidden Markov processes simplify the hidden and observed probabilities to

[*]e-mail: glinport@isu.edu

just one transitional probability. The process is iterated to generate a full sequence of tokens. The simplicity of these models lends to them being easy to implement and efficient to run [12]. Examples of Markov implementations include systems that react interactively to music input [11] and text-to-speech synthesis where speech waveform generation is generated via a hidden Markov model [13].

When generating sequences, Markov processes produce sequences that share a common style with the data the model was trained on. In the context of natural language synthesis, the style sharing properties of Markov processes can be exploited to change speaker identities, emotion of the speech, and the cadence of the speaker [13].

Previous efforts have been made to combine probabilistic models with constraint satisfaction. Pachet et al. introduce constrained non-hidden Markov process as a method for applying user-defined constraints to a non-hidden Markov model [12].

A previous visualization[1] (see Figure 1) was created to visualize a constrained non-hidden Markov process as described by Pachet et al. [12]. This previous work is limited in only visualizing a constrained *non-hidden* Markov process rather than a more complex constrained *hidden* Markov process. The visualization only shows a small subset of inner model connections and nodes and lacks user interaction to explore the rest of the model.

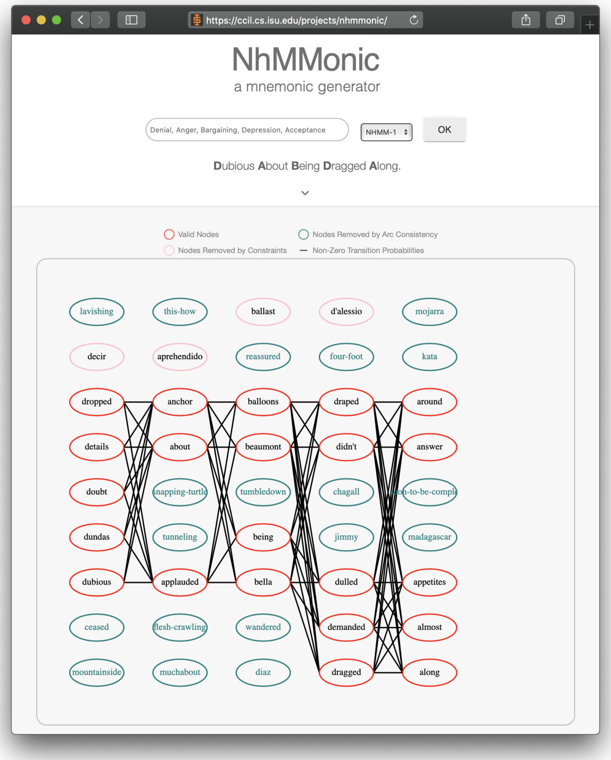

Figure 1: Website view of a previous visualization of a constrained *non-hidden* Markov process [1]. Notably, the visual is a two dimensional view of a portion of the model (i.e. only a small subsection of nodes and words are shown). Also note that a *non-hidden* Markov process is structurally less complex than a *hidden* Markov process which has both hidden nodes and observed nodes.

In subsequent work, Pachet's method is expanded to apply user-defined constraints to hidden Markov processes [5]. This is the model that will be visualized in this paper.

At a high level, a constrained hidden Markov process is created by first training a non-constrained hidden Markov process. That trained model is copied for each position of a finite sequence. User-defined constraints, such as the first word of a sequence needs to rhyme with "red", are applied to relevant sequence positions, which involves removing (or setting probabilities to 0.0) transitions to non-satisfying nodes. Then the entire series of transitions are traversed to check for arc-consistency, meaning that transitions to non-satisfying solutions are removed. The last, and most crucial, step in the process is to re-normalize all probabilities such that the original probability distribution remains intact. Further detail will not be discussed in this paper but can be found in papers referenced [5] [12].

3 Methods

3.1 Data

Data was retrieved from an implementation of a constrained hidden Markov process hosted on github[2]. The implementation was modified to output internal model information. Data retrieved includes hidden node name, status, and transition information. Hidden node status pertains to if the node was removed during constraint satisfaction or during the enforcement of arc-consistency. Observed node name and transition information is also retrieved. Data is put into a JSON file that heavily utilizes the array function of the JSON spec.

3.2 Visualization implementation

The visual is implemented in a web environment using a the WebGL library: ThreeJS. Using a WebGL library over data visualization libraries such as D3.js (using D3-3D.js) or the graph network focused tool Gephi due to the flexibility and control afforded by working with next-to-low-level object. By using ThreeJS rather than pure WebGL abstracted many laborious tasks away from the 3D visualization such as writing shaders.

The visualization is a graph-based visual and thus sphere objects are created for nodes and Bézier curves are used for the edges between nodes. ThreeJS allows for these objects to be created with object properties such as material, size, and position in 3D space. Text sprites are created using an extension of ThreeJS for convenience[3]. Animations are implemented using a javascript animation suite called TweenMax.

The implementation of the model as well as example data can be found at the following github repository: `https://github.com/po-gl/ChimpVisual`.

3.3 Relation to Tufte's Guidelines

During the designing process of the visualization, Tufte's guidelines for visual excellence were consulted [8]. In striving to maximize data-ink ratios, the visualization takes on a minimalistic appearance to avoid visual distractions typical of poor 3D visualizations. Visually distracting elements such as shading and unnecessary outlines are avoided. The lighting of the model is flat with no shading on node spheres. Text is displayed as sprites rather than overly complicated 3D objects themselves and are a solid and easy to read font (Times New Roman).

Another of Tufte's guidelines, aesthetic elegance, is a motivation for the visual's design. Rather than edges being straight from node to node, edges are implemented as Bézier curves which are more aesthetically pleasing and help to reduce edge clutter. Hidden nodes are very slightly transparent which lets the viewer recognize there is content behind the node but remains opaque enough to clearly read

[1]Website for previous visualization available here: `https://ccil.cs.isu.edu/projects/nhmmonic/`.

[2]Constrained hidden Markov process implementation hosted here: `https://github.com/brandonbiggs/Chimp`.

[3]Text sprite extension can be found here: `https://github.com/SeregPie/THREE.TextSprite`

text on the foremost node. Elements of the visualization are animated such as switching between selected hidden node word clouds and lowering the opacity of word clouds to de-focus them—de-focusing visual elements also benefits a viewers ability to pre-attentively process the visual or take in the visual without mental strain. Rather than elements popping in and out, subtle and fast animations are added to make the experience of exploring the visual as smooth as possible. Cutting out any jarring elements in the visual is done in the name of increasing the aesthetic elegance of the visualization.

4 Results

The resulting visualization is shown in Figure 2. This model is trained on a toy-sized dataset (4 sentences) for illustrative purposes—the toy-sized model example is used and described further in a referenced paper [5]. The model is constrained such that the last word in the sequence is "red" and the first word rhymes with "red" The camera is placed above the model and is slightly skewed. Green nodes represent selected hidden nodes, dark grey represent nodes removed during constraint satisfaction, and light grey represent nodes removed during the enforcement of arc-consistency. White nodes represent non-selected hidden nodes that are not removed in the model; however, this toy example has all non-removed nodes selected already. The graph-based visual shows four clear columns of hidden nodes that represent the parts-of-speech allowable for that position in the sequence. For example, we can see that the first position has three dark grey nodes and "Proper Noun" selected. Black edges represent allowable transitions between hidden nodes creating a path in the visual from "Proper Noun" to "Adverb" to "Verb" to "Noun". Underneath hidden nodes are where observed nodes are displayed as word clouds. The opacity of the word clouds is lower due to the camera focusing on the hidden nodes above.

In this toy example, we can see that each sequence position only has one allowable part-of-speech and thus the word selection is quite restricted. This makes sense due to the very small training set as there are very few examples of word transitions to learn from.

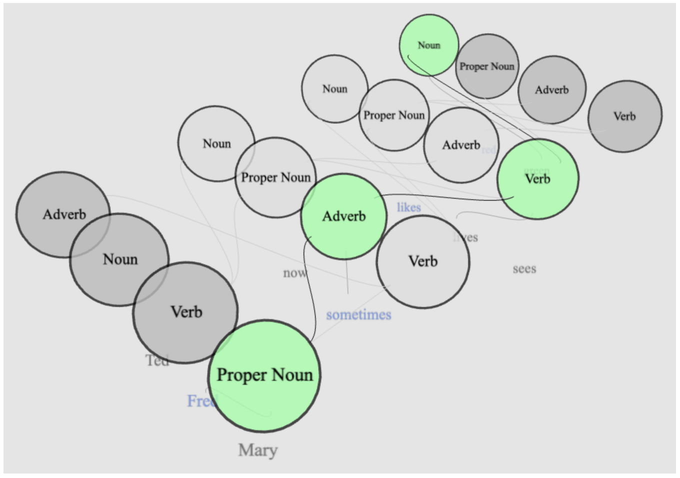

Figure 2: Visualization of a constrained hidden Markov process. Node circles represent the hidden nodes of the model with edge connections between nodes. Green nodes represent the selected hidden node for which the observed node "word cloud" is displayed below it. Dark grey nodes represent hidden nodes that were removed during constraint satisfaction. Light grey nodes represent hidden nodes that were removed during the enforcement of arc-consistency.

In Figure 3, we can see the same toy example visualized from a lower camera angle. Note that now the word cloud opacity is raised due to the camera now focusing on the observed node word clouds (the opacity is changed by a subtle and fast animation). Each word cloud is displaying usable words that the model could use from the hidden node selected above the word cloud. Words in blue represent a particular sequence solution that the model could output as an artifact: "Fred sometimes likes red".

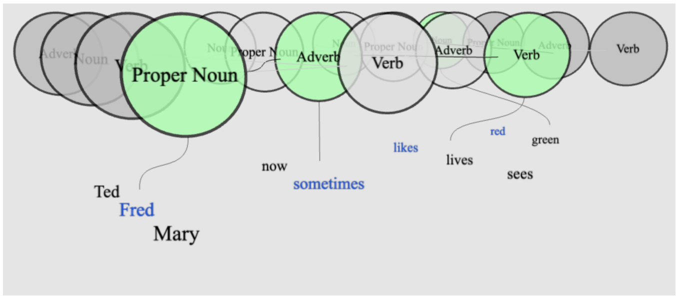

Figure 3: The observed nodes in a constrained hidden Markov process are visualized as "word clouds" underneath selected hidden nodes.

Finally, in Figure 4, we see a visualization of a model trained on a more substantial dataset (100 sentences from the COCA dataset [3]).

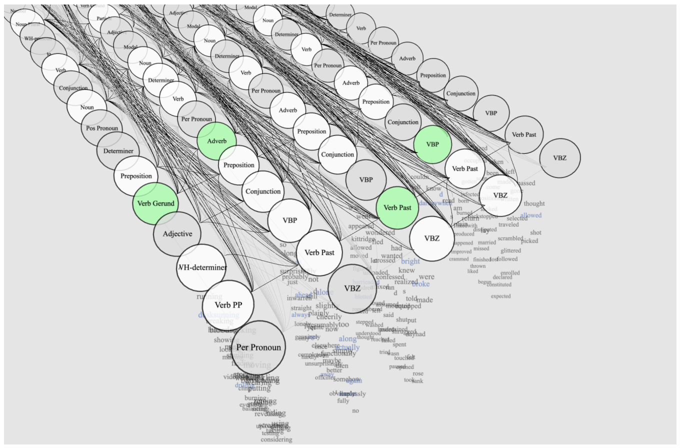

Figure 4: The visualization becomes complicated when visualizing a constrained hidden Markov process trained on 100 sentences (using COCA dataset [3]). However, the visualization does still convey some information to viewers. In particular, we can see by the number of grey nodes that the last sentence will be very restricted in which word can be chosen.

5 Discussion

The visual itself inherently displays more information about the model and should inherently lead to the increased perception of system intentionality to an observer. This increase in perceived intentionality lends to the system being better able to exhibit behaviors an observer might perceive to be more creative; in other words, increased intentionality leads to a better creative system. The extent to which intentionality or explainability of a constrained hidden Markov process by this visualization is to be determined in future work.

The visualization itself is somewhat limited in what information is displayed. While model inner connections and node removal information is displayed, the actual model transition probabilities are absent in the visualization. Displaying transition probabilities could further increase what model information is shown and could increase the perceived intentionality of the model. Another piece of model information absent is mathematical values generated during renormalization; this information could be displayed alongside other probabilities to further illustrate to observers that the model is generating artifacts in a complex and non-random way.

When visualizing a substantially trained model, see Figure 4, the visualization suffers from becoming visually cluttered. However, there still is valuable information about the model being displayed. For example, an observer can see that the last position in the sequence is mostly grey which indicates that the last word will be heavily restricted in what the model can choose for it.

6 Conclusion and Future Work

Through the use of the WebGL library ThreeJS, model information from a constrained hidden Markov process is displayed in a three dimensional visualization. The visual shows inner model connections that lend to an increased perception of intentionality and explainability of the constrained hidden Markov process to an observer.

Future work involves displaying model probabilities by way of directly placing them on nodes and edges or through a mouse hovering interaction. Future work also involves making visualizations of substantially trained models less visually cluttered. Solutions to de-clutter a complicated visual could be to show a detailed view of a subsection of the model with a low-detail ”thumbnail” view of the entire model below; allowing users to change subsections by interacting with the low-detail entire model view.

The extent to which intentionality and explainaiblity is increased for the constrained hidden Markov process due to the visualization can also be explored. A qualitative survey could be conducted in which participants are asked their opinions on intentionality when shown the visualization or artifacts alone. Regardless of future work possibilities, the visualization as it stands provides a way to quickly gain some understanding of a constrained hidden Markov process.

References

[1] P. M. Bodily, P. Glines, and B. Biggs. “She Offered No Argument”: Constrained Probabilistic Modeling for Mnemonic Device Generation. In D. V. M. L. M. Kazjon Grace, Michael Cook, ed., *Proceedings of the 10th International Conference on Computational Creativity*, pp. 81–88. Association for Computational Creativity, Charlotte, North Carolina, 2019.

[2] S. Colton, G. A. Wiggins, et al. Computational creativity: The final frontier? In *Ecai*, vol. 12, pp. 21–26. Montpelier, 2012.

[3] M. Davies. The 385+ million word Corpus of Contemporary American English (1990–2008+): Design, architecture, and linguistic insights. *International Journal of Corpus Linguistics*, 14(2):159–190, 2009.

[4] P. Glines, B. Biggs, and P. M. Bodily. A leap of creativity: From systems that generalize to systems that filter. In K. Grace, M. Cook, D. Ventura, and M. L. Maher, eds., *Proceedings of the 11th International Conference on Computational Creativity*, pp. 297–302. Association for Computational Creativity, 2020.

[5] P. Glines, B. Biggs, and P. M. Bodily. Probabilistic generation of sequences under constraints. In *2020 Intermountain Engineering, Technology and Computing (IETC)*, pp. 1–6, 2020. doi: 10.1109/IETC47856.2020.9249157

[6] M. T. Llano, M. d’Inverno, M. Yee-King, J. McCormack, A. Ilsar, A. Pease, and S. Colton. Explainable computational creativity. In *Proceedings of the Eleventh International Conference on Computational Creativity, ICCC*, 2020.

[7] M. Loller-Andersen and B. Gambäck. Deep learning-based poetry generation given visual input. In F. Pachet, A. Jordanous, and C. León, eds., *Proceedings of the Ninth International Conference on Computational Creativity (ICCC’18)*, pp. 240–247. Association for Computational Creativity, Salamanca, Spain, 2018.

[8] T. Munzner. *Visualization analysis and design*. CRC press, 2014.

[9] A. Nayebi and M. Vitelli. Gruv: Algorithmic music generation using recurrent neural networks. *Course CS224D: Deep Learning for Natural Language Processing (Stanford)*, 2015.

[10] A. OÃ±ate, G. MÃ©ndez, and P. GervÃ¡s. Emolift: Elevator conversations based on emotions. In K. Grace, M. Cook, D. Ventura, and M. L. Maher, eds., *Proceedings of the 10th International Conference on Computational Creativity*, pp. 124–131. Association for Computational Creativity, Charlotte, North Carolina, USA, 2019.

[11] F. Pachet. The Continuator: Musical Interaction With Style. *Journal of New Music Research*, 32(3):333–341, sep 2003. doi: 10.1076/jnmr.32.3.333.16861

[12] F. Pachet, P. Roy, and G. Barbieri. Finite-length Markov processes with constraints. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, 2011. doi: 10.5591/978-1-57735-516-8/IJCAI11-113

[13] K. Tokuda, Y. Nankaku, T. Toda, H. Zen, J. Yamagishi, and K. Oura. Speech synthesis based on hidden Markov models. *Proceedings of the IEEE*, 101(5):1234–1252, 2013. doi: 10.1109/JPROC.2013.2251852

[14] D. Ventura. Mere generation: Essential barometer or dated concept? In F. Pachet, A. Cardoso, V. Corruble, and F. Ghedini, eds., *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2016)*, pp. 17–24. Sony CSL, Paris, France, June 2016.

[15] D. Ventura. How to Build a CC System. In *Proceedings of the Eighth International Conference on Computational Creativity*, pp. 253–260, 2017.