Competing Visions for AI

Turing, Licklider and Generative Literature

Oscar Schwartz

Abstract

In this paper, I will investigate how two competing visions of machine intelligence put forward by Alan Turing and J. C. R Licklider – one that emphasized automation and another that emphasized augmentation – have informed experiments in computational creativity, from early attempts at computer-generated art and poetry in the 1960s, up to recent experiments that utilise Machine Learning to generate paintings and music. I argue that while our technological capacities have changed, the foundational conflict between Turing's vision and Licklider's vision plays itself out in generations of programmers and artists who explore the computer's creative potential.

Moreover, I will demonstrate that this conflict does not only inform technical/artistic practice, but speaks to a deeper philosophical and ideological divide concerning the narrative of a post-human future. While Turing's conception of human-equivalent AI informs a trans-humanist imaginary of super-intelligent, conscious, anthropomorphic machines, Licklider's vision of symbiosis underpins formulations of the cyborg as human-machine hybrid, aligning more closely with a critical post-human imaginary in which boundaries between the human and technological become mutable and up for re-negotiation. In this article, I will explore how one of the functions of computational creativity is to highlight, emphasise and sometimes thematise these conflicting post-human imaginaries.

Let base clouds stir the world's enshrouded tears
Which have no astronomy to be assail'd
Thus in thy fair appearance lay thy buried fears
Whose uncovering gaze my fond perception failed

(Matias 2010: n. p.)

DOI 10.14361/dcs-2018-0107

PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away these miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

(Karpathy 2010: n. p.)

The two poetic texts above are products of experiments in generative literature. The first is a poem created by J. Nathan Matias' Swift-Speare, a generator that uses machine learning software and a predictive text framework called TouchType, generally used for text messaging (Matias 2010: n. p.). The second is blank verse created by Andrej Karpathy's Recurrent Neural Network (RNN) trained on a corpus of Shakespearean text.

Both Matias and Karpathy's poetry generators use machine learning, a category of Artificial Intelligence (AI) software that can complete tasks without programmers having to think through in advance the steps necessary to solve the problem (Mitchell 1997: xv). That is, with machine learning systems, one provides a goal for the system to achieve, and then trains it on a large amount of data, wherein the system learns to solve the problem on its own (Mitchell 1997: xvi). Loosely inspired by the neural networks of the human brain, the system appears to "learn" from experience, and becomes better at tasks through repeating this learning process (Mitchell 1997: 82). Although machine learning technology has been discussed since the 1950s, it is only in the last decade, with advancements in a new type of machine learning called deep learning, that it has been effectively implemented to perform many tasks previously considered beyond computer aptitude, including image recognition and natural language processing (Marr 2016: n. p.). According to experts, these recent developments in deep learning have "revolutionized the field and AI and represent a step towards building autonomous systems" (Arulkumaran et al 2017: 1). Indeed, it is recent advancements in machine learning that have led to an upsurge in interest in the future of AI in the technology sector, in academic research, and in the public imaginary. In recent public talks, Elon Musk has perhaps best captured the prevailing sentiment of this futurist, innovation centric moment, proposing that new developments in machine learning will lead to the development of robots and systems that will not only replace humans as

workers, but will transform us and the world entirely, potentially usurping humans as a predominant species on earth (Bogost 2017: n. p.).

It is important to note, however, that while both Matias and Karpathy use machine learning systems in their literature generating experiments, they articulate very different attitudes and approaches to how these systems should be implemented. For Matias, the machine learning system used to generate Shakespearean poetry is conceptualised as a partner in the creative process, a collaborating agent in a productive relationship with Matias. On the other hand, for Karpathy, the RNN is conceptualised as an agent of automation, a system to which literature generation is outsourced, optimised and accelerated.

In this paper, I argue that these two conflicting conceptualisations of machine learning can be traced back to two conflicting visions of AI formulated in the inaugural decade of research into machine intelligence. To do so, I will examine formative papers written by two AI pioneers – Alan Turing and J. C. R Licklider – who each articulate different visions for the field's future. I will demonstrate that Turing's vision proposed that the act of making a machine intelligent requires transferring anthropomorphic qualities to the computer, and fosters the idea of computers automating and replacing humans – I will call this the automative vision of AI. By comparison, Licklider's vision maintained that machine intelligence can be harnessed through human-machine collaboration; I call this the hybrid vision of AI. I will also demonstrate that these two formative visions of AI speak to a deeper philosophical and ideological divide concerning the narrative of a post-human future. While Turing's conception of human-equivalent AI informs a transhumanist imaginary of super-intelligent, conscious, anthropomorphic machines, Licklider's vision of symbiosis underpins formulations of the cyborg as human-machine hybrid, aligning more closely with a critical post-human imaginary in which boundaries between the human and technological become mutable and up for re-negotiation

In this historically oriented paper, I propose that Turing and Licklider's visions of machine intelligence are instantiated by and become acutely visible in two different approaches to generating literature with computers.[1] In one approach, the programmer attempts to program creative agency into the computer by enabling a functional mirroring of human poetry. In the second approach, the programmer collaborates with the computer to create a new hybrid form of creative agency that

1 Generative literature has been the focus of a number of historical studies, though it is often referred to as computer-generated literature. In his historical survey, *Pre-Historic Digital Poetry,* Christopher Funkhouser defines computer-generated literature as texts "generated by computer algorithm, arranged as a sequence of words or signs and symbols according to programming code" (2007: 31). He further claims that this is the "archetypal" genre of digital literature, and that, in fact, programming of computers for text generation can be traced back to the inaugural moments of computer science (2007: 36).

facilitates an exploration of novel forms of literature-making. To interrogate this claim, I will look at historical examples of both approaches to literature generation developed in the 1980s and 1990s by Ray Kurzweil and Charles Hartman, drawing out correspondences between their practices, and the visions of Turing and Licklider.

Having done so, I will return to a discussion of Matias and Karpathy, demonstrating that their diverging approaches to using machine learning for generating literature can be productively understood as a continuation of this historical conflict between the android and the cyborg, the automative and the hybrid. Thus, at the broadest level, the comparison of these three linked moments in the history of AI and generative literature – Turing/Licklider, Kurzweil/Hartman, Karpathy/Matias – brings into focus continuities in technological and cultural history. While much of the conversation about machine learning concerns disruption and the future, my aim is to use generative literature as a way contextualise recent developments in AI within a historical perspective. This is an important task within cultural critique of technology, as it tempers the default techno-utopian, technological determinist perspective that fetishizes new AI technologies, and in so doing open our eyes to alternative conceptions of machine intelligence, beyond Elon Musk's predictive and determinist framework, offering analysis that broadens and deepens the critical discussion of the intersections between AI as technical pursuit and its cultural implications. The historical approach, I will show, allows us to develop modes of critique in the present, and makes the future appear less technologically determined, more a matter of conflict and debate at the intersection of technology and culture.

To begin this analysis, I will return to the decade when research into intelligent computers began. While the dream of creating intelligent machines is "an ancient wish", the field of Artificial Intelligence – or the particular pursuit of making digital computers intelligent – has its origins in the 1950s (McCorduck et al 1977: 951–2). Bookending this decade are two papers that articulate two visions for AI's future: Alan Turing's "Computing Machinery and Intelligence" (1950), and J.C.R Licklider's "Man-Computer Symbiosis" (1960). Underpinning these different visions is not only a difference of opinion about the future of technology, but also diverging conceptions of the narrative of our post-human future.

Two Visions for Machine Intelligence

After the hardware-engineering problem of building the first digital computers had been overcome in the mid-1940s, Alan Turing turned his focus to discovering what range of human activities could be translated into machine-interpretable algorithms, and thus performed by the computer (Turing 2013: 401).[2] The most

2 The conception of the digital computer as programmable machine was articulated by Alan Turing in his 1936 paper "On Computable Numbers". In this paper, Turing conceives of an automatic computing machine – a Turing machine – which is a

obvious human intelligence tasks that could be computerised were those that were already governed by formalised methods, like calculating bomb or rocket trajectories (Turing 2013: 23).[3] However, Turing also began to explore the idea that a more diverse range of human behaviour could be performed by the computer. While held back by limitations in speed and processing in his time, Turing predicted that machine intelligence would lead to the automation of a diverse range of human tasks. He wrote, "I do not see why it [a computer] should not enter any one of the fields normally covered by the human intellect, and eventually compete on equal terms. I do not think you can even draw the line about sonnets [...]" (2013: 358).

Developing this line of thought, in 1950 Turing published "Computing Machinery and Intelligence", in which he poses the question, "Can machines think?" Rather than defining the terms "machine" and "think", Turing outlines another closely related question derived from a Victorian parlour game called the imitation game (Turing 1950: 433). The rules of the imitation game stipulate that a man and a woman, in different rooms, communicate with a judge via handwritten notes, answering the judge's questions. Reading these answers, the judge has to determine who is who. The judge's task is complicated by the fact that the man is trying to 'imitate' the woman in order to lead the judge into the wrong identification (1950: 433–434). Turing postulated a scenario in which the male contestant in the game was substituted for a computer. If this computer was programmed to play the imitation game so that an average interrogator would have no more than 70 per cent chance of making the right identification after five minutes of questioning, then it would be reasonable, Turing argued, that the machine could be said to possess intelligence (1950: 442). That is, Turing suggests that the question, "Can machines think?" should be replaced by, "Are there imaginable digital computers which would do well in the imitation game?" (1950: 442).

Embedded in "Computing Machinery and Intelligence" are two key ideas: first, programming is an activity that the human does to the computer in order to invest it with a type of intelligence. For example, one might develop an algorithm

simple theoretical device, or model, that can manipulate symbols according to a set of instructions (Turing 1938: 117). The theoretical breakthrough Turing makes in this paper is that one does not have to build a new machine to perform some computable task, but only abstractly describe it to the machine with a program. That the machine is programmable, Turing realised, makes it "universal". In other words, a universal machine can perform any task whose method can be formalised in an algorithm. The universality of this computing machine is afforded by its capacity to be programmed to behave identically to other machines through the interpretation of encoded descriptions.

3 Turing and other pioneering computer scientists realised that an appropriately programmed stored-program computer could replace the thousands of people formally employed to routinely execute these calculations. Turing wrote: "a good working rule is that the [digital computer] can be made to do any job that could be done by a human computer, and will do it in one tenthousandth of the time." (Turing 2013: 378)

for playing chess, and script it as a program for the computer, thus endowing the computer with the ability to play chess; chess-intelligence is transferred from the human to computer through the act of programming.[4] Second, the computer is deemed to be proficient or intelligent if it is able to perform the activity indistinguishably from a human actor. That is, the computer has chess playing capacity if it can functionally mirror a human chess player. By extension, assessing the intelligence of the computer becomes an act of normative judgment in which the human assesses the computer's behaviour against an existing human standard.

Turing's ideas, as outlined above, formed an influential vision for AI that influenced a generation of pioneering computer scientists after his death, including Alan Newell, Herbert A. Simon and Marvin Minsky, among others, who were dedicated to replicating intelligent human behaviour in the computer, including playing games, proving theorems, and generating language (Floridi 1999: 142–146).[5] Underpinning these activities was the Turing-inspired principle that if a computer could imitate the behaviour of another actor, then regardless of the precise inner workings of their internal operations, these two actors could be considered equivalent (Floridi 1999: 133). Emerging from this principle was a belief that in the not-to-distant future, there would android AIs, capable of replicating human actors in all tasks. For example, in 1965 Simon wrote: "machines will be capable, within twenty years, of doing any work a man can do" (Crevier 1995: 109).

This vision of the anthropomorphically intelligent AI subsequently infiltrated and was developed in broader cultural discourses: philosophers including Daniel Dennett, John Searle and Hubert Dreyfus debated whether human-level intelligence could be replicated in the machine[6]; novelists like Stanisław Lem, Arthur C. Clarke and Phillip K. Dick explored the implications of anthropomorphic AI in their science fiction narratives.[7] The focus of this discourse was distinctly posthuman, exploring ideas of boundary disturbance elicited by the figure of the intelligent machine. Specifically, the idea of anthropomorphic AI blurred boundaries between the human and the machine in two distinct ways: firstly, the intelligent

4 In the late 1940s, Turing imagined an experiment in which an average chess player played against another average human player, and then against a chess algorithm. Turing proposes that if the player could not tell the difference between which one they were playing, the algorithm works (Turing 2013: 431).

5 This paradigm became known as Good Old Fashioned Artificial Intelligence, or GOFAI, which dominated the landscape and informed the research methodologies in the years after Turing's death. (Floridi 1999: 132): See, for example, Newell, Shaw and Simon, "Empirical Explorations with the Logic Theory Machine: A Case in Heuristics"; Newell and Simon "Computer Science as Empirical Inquiry: Symbols and Search".

6 See Daniel Dennett, "Can Machines Think"; John Searle, "The Chinese Room Argument"; Hubert Dreyfus, *What Computers Can't Do*.

7 See Stanisław Lem, *Summa Technologiae*; Arthur C. Clarke, *2001: A Space Odyssey*; Phillip K. Dick, *Do Androids Dream of Electric Sheep?*

computer suggested that as computers became more sophisticated they would become more human: this in turn led to the figuration of the android AI, a fully human machine, that becomes conscious of itself, and suggests a transcendence of ontological boundaries between the human and technological (Suchman 2007: 226–228). Secondly, the programmable intelligent machine suggested that all human capacity might be programmable, and that the human is simply a complicated information processor. As David Golumbia explains, within this discourse "our notion of intellect is, at bottom, identical with abstract computation, and that in discovering the principles of abstract computation via the Turing Machine human beings have, in fact, discovered the essence not just of human thought in practice but all thought in principle" (2009: 7). This idea in turn unsettles boundaries between the human and the machine, not by figuring the machine as humanised, but by proposing that humans are, fundamentally, machines.[8]

As can be seen, the vision developed by Turing in his early papers on computing technology and machine intelligence had a lasting influence both on AI as a technical pursuit and on the post-human discourses surrounding it. However, an alternative vision was developed a decade after Turing published "Computing Machinery and Intelligence", in J. C. R Licklider's 1960 paper "Man-Computer Symbiosis". In this paper, Licklider offers a vision of machine intelligence that focuses on human-machine collaboration. He begins by outlining two shortcomings of the automative vision of machine intelligence proposed by Turing.[9] Firstly, Licklider argued that by instructing computers to imitate activities previously performed by human actors, computers would be limited to activities previously performed by humans, without consideration of novel previously unimaginable intellectual activities that the technology offers (Licklider 1960: 4). Secondly, Licklider proposed that the automative paradigm presupposes a theoretical equivalence of human-computer capacity; however, there are "certain genotypic differences in the capability between men and computers", and that "computing machines can do readily well, and rapidly many things that are difficult or impossible for man, and men can do readily well, though not rapidly, many things that are difficult or impossible for computers" (1960: 6). Having outlined the shortcomings of the automative vision of machine intelligence, Licklider presents his own vision of "an intuitively guided trial-and-error procedure" that requires the "cooperative interaction between men and electronic computers" and a "very close coupling between the human and the electronic members of the partnership" that "enable[s] men and computers to cooperate in making decisions and

8 The idea of the human being "mere machine", of course, does not emerge with Turing, but can be traced back to the materialist philosophies of the Enlightenment, in particular Julien Offray de la Mettrie's *L'homme machine*. See Isaiah Berlin, *The Age of Enlightenment*.

9 Licklider does not specifically mention Turing, but he mentions some of his predecessors, including Simon (1960: 5).

controlling complex situations without inflexible dependence on predetermined programs" (1960: 1, 5). Underpinning this vision are three main principles. First, in the collaboration between human and machine, humans perform tasks that humans are proficient at, and machines perform tasks that machines are proficient at: humans "set the goals, formulate hypotheses, determine criteria, and perform evaluations"; computers "do the routinizable work" and "interpolate, extrapolate, and transform" (1960: 7). Licklider's second principle is that collaboration improves the capacity of both human and computer intelligence through their symbiosis, allowing for new, previously unimaginable "intellectual operations" (1960: 1). As Licklider writes:

> The hope is that in not too many years, human brains and computing machines will be coupled together very tightly, and that the resulting partnership will think as no human brain has ever thought and process data in a way not approached by the information-handling machines we know today. (Licklider 1960: 4)

Licklider's final principle is that, in their interaction, the human and computer become a symbiotic assemblage. That is, the programmed computer is not a "mere extension" of the human, or an imitation of the human, but a signifier of a cooperative "living together in intimate association, or even close union, of two dissimilar organisms" (1960: 4).

Unlike Turing's automative vision of machine intelligence, Licklider's vision is underscored by principles of hybridity: it requires dynamic and modular collaboration between human and nonhuman actors; the authority of individual contributors in the productive act are de-emphasised, and the collaborative contributions effectively combined; the collaboration enables the exploration of new intellectual activities, rather than the imitation or automation of previous ones; and finally, agency is not transferred from the human to the machine, but a new hybrid form of agency is created through human-machine symbiosis.

These principles of an augmented machine intelligence through human-computer interaction were influential, and further developed by a number of computer pioneers including Douglas Engelbart, Alan Kay and Seymour Papert: each dedicated themselves to designing systems for dynamic moments of hybridity between human and machine intelligence, rather than trying to endow the machine with anthropomorphic agency.[10] These visions of human-machine symbiosis were also developed in a broader cultural context by philosophers and theorists like Donna Haraway, Lucy Suchman, and N. Katherine Hayles.[11] Rather than imagining android machines replacing humans, these theorists explore

10 See Douglas Engelbart, "Augmenting Human Intellect: a conceptual framework"; Seymour Papert, *Mindstorms: Children, computers and powerful ideas*.

11 See Donna Haraway, *Simians, Cyborgs, and Women*; N. Katherine Hayles, *How We Became Post-Human*; Lucy Suchman, *Human-Machine Reconfigurations*.

ideas of boundary disturbance elicited by the figure of the cyborg. Haraway, for instance, argues that a cyborg exists when boundaries between "self-controlled, self governing machines (automatons) and organisms, especially humans (models of autonomy)" are "problematic"; in other words, the cyborg is a boundary-blurring "figure born of interface of automaton and autonomy" (Aarseth 1997: 54).

As can be seen, the competing visions for machine intelligence articulated by Turing and Licklider a decade apart loomed large over subsequent attempts at building intelligent systems as well as informing discussions about the post-human implications of machine intelligence. While there are many ways to track the influence of these two papers, an effective way of doing so is looking at two different approaches to generating literature with computers developed in the following decades. Specifically, I propose that these two visions inform and are instantiated in two approaches to generative literature developed in the 1980s and 90s by Ray Kurzweil and Charles Hartman. As I will demonstrate in the following section, in Kurzweil's approach, the programmer attempts to program poetic agency into the computer by enabling it to imitate pre-existing forms of human poetry; the computer becomes the poet through a functional mirroring of existing poetry, and reading the output text often becomes a type of Turing-like test, a normative act in which the reader attempts to judge whether it was written by a human or a computer. This approach to generative literature corresponds to Turing's vision of machine intelligence, and the associated discourses concerning anthropomorphic AI. By comparison, in Hartman's approach to generative literature, the human programmer collaborates with the computer in the act of literature-making; creative agency arises from human-machine symbiosis; and the literature produced is not judged according to the verisimilitude to human poetry, but its novelty or capacity to bring forth new forms of literary activity and language. This approach to generative literature corresponds to Licklider's vision of machine intelligence, and the associated discourses of the cyborg. In the following readings, I will draw out the correspondences and continuities between Turing and Licklider's visions, and Kurzweil and Hartman's approach to generating literature, not only at the level of formal techniques, but also in terms of how their literary experiments ramify within broader post-human discourses. I will also highlights points of difference as technological principles were put to use in aesthetic practice, which will allow me to demonstrate how technology informs cultural practice, and how, in turn, cultural practice can extend technological principles.

Kurzweil and Hartman: Two Approaches to Generative Literature

From the 1980s to the 1990s, Ray Kurzweil developed an algorithm for generating poetry called RKCP that generated poetry according to statistical modelling and n-gram generation (Kurzweil 2000: n. p.). The way this process works is as follows: first, RKCP is shown a corpus of poems written by a poet or poets: in general,

Kurzweil uses famous poets like Emily Dickinson or Robert Frost (Kurzweil 1999: 164). Second, RKCP performs a statistical analysis of the language used in this corpus of poetry, sorting the poems into words, word structures, rhythm patterns and poem structure – this is what Kurzweil calls a "language model" (Kurzweil 1999: 163). Third, using the "language model", RKCP generates a poem that statistically approximates the poetry written by the human poet. The output is a poem that resembles, but does not plagiarise, the poetry RKCP was originally shown (Kurzweil 1999: 163). For example, after being trained on a corpus of Emily Dickinson poems, RKCP generates the following poem:

> Rain is a star,
> Some birds spoke out for you
> about the tale of sages (Kurzweil 2001: n. p.)

In order to test the success of RKCP, Kurzweil conducted two Turing-like tests, in which he gave 28 poems to 16 human judges, both adults and children (Kurzweil 1999: n. p.). Of the 28 poems, 12 were written by human poets, and the others were generated by RKCP. The 13 adult judges guessed the provenance of the poems correctly 63 percent of the time, while the three child judges scored an average of 48 percent. In the second iteration of the test, a performer read both RKCP poems and the human poems to a crowd. Kurzweil reports, "the audience was able to correctly identify whether a particular poem was written by a human or by the computer less than 50 % of the time" (Kurzweil 2000: n. p.).

Kurzweil's approach to developing and testing RKCP instantiates Turing's vision for machine intelligence in a number of significant ways: firstly, Kurzweil's practice constitutes the act of programming a computer to produce strings of output in imitation of human poetry. In fact, Kurzweil refers to language models that RKCP makes from the corpus texts of human poets as "poet personalities", as if the language model itself is a disembodied realisation of Dickinson's poetic identity (Kurzweil 2000: n. p.). Secondly, in his brief analysis of the results of his Turing-style test of RKCP's poetry, Kurzweil postulates that RKCP's success is validated by its ability to achieve a degree of indistinguishability from human poetry in the Turing-style tests: for example, although Kurzweil acknowledges that RKCP is not yet a fully anthropic poetic agent, he proposes that its success in the Turing test foreshadows a point in the future – 2029 to be precise – when "many of the leading artists are machines" (Kurzweil 1999: 223). An underlying implication is that to model and regenerate the language of a poet, is to emulate and, through that emulation, possess an equivalent form of creative intelligence.[12]

12 N-gram language generation has its origins in Claude Shannon's paper "A Mathematical Theory of Communication". His key insight was that the English language, and in fact any language, has a degree of statistical predictability, meaning that it can be simulated via probabilistic generation. As such, one can take a given corpus of

Moreover, Kurzweil's automative approach to generative literature is an expression of his broader visions about AI's future, which closely resemble, yet also extend Turing's. For example, in *The Age of Spiritual Machines*, Kurzweil expresses the view that if a computer can be programmed to imitate the behaviour of another actor, then regardless of their ontological status, these two actors can be considered equivalent: it should not matter whether the output is made by human or computer, only whether the output is fit for function (1999: 61–63). Whether in chess, diagnosing medical conditions, buying and selling stock, or guiding cruise missiles, one can test the capacity of the computer by using human intelligence as a template against which to judge the computer's aptitude (1999: 277–280). If the product is sufficient, the question of provenance becomes irrelevant, or even worse, species-chauvinistic (Kurzweil 2012: 144). Moreover, like Turing, who claimed that the computer could enter into any one of the fields normally covered by the human intellect, and eventually compete on equal terms, Kurzweil holds that all human activities are programmable, and will be replicated by computers. However, Kurzweil's experiment in computer-generated poetry does highlight a considerable difference between his and Turing's conception of machine intelligence. For Turing, his test was a way of assessing *general intelligence*; that is, for a computer to pass the test it would require the capacity to speak convincingly on all topics, including poetry, chess, mathematics, small talk, etc. Merely mastering one type of skill, therefore, did not indicate that the computer was closer to this general intelligence, which required flexibility and adaptability. Indeed, the inherent difficulty in acquiring the type of intelligence required to pass the generalised test has led many theorists to assert that Turing's test was indeed a philosophical provocation rather than a normative framework.[13] For Kurzweil, on the other hand, the Turing test is a methodology that can be used to test a computer's *progression* towards general intelligence. That is, if a computer can pass a number of domain-specific Turing tests, then it is demonstrating, for Kurzweil, increasing intellectual capacity. This notion fits within Kurzweil's broader conception of the singularity, the idea that computing power accumulates over time towards a point of completion (Kurzweil 2000: 270–280). Thus, RKCP, although a relatively simple poetry-generating algorithm, is, for Kurzweil, a stepping-stone on the way to this point in the future. Turing does not, at any point, imply that there is this type of linear progression towards general intelligence. In this sense, Kurz-

language and describe it via a language model, which outlines the chance of certain words being used in combination. Language is treated as raw material, organised statistically instead of semantically. Shannon explains that, "a sufficiently complex stochastic process will give a satisfactory representation of a discrete source" (386). The implication for a program like RKCP is that as the stochastic process improves and approximates itself probabilistically to the poetic language of whichever corpus you choose to generate, it will provide a more satisfactory result.

13 See Graham Oppy and David Dowe "The Turing Test".

weil's application of a Turing-like principle to generative literature demonstrates how Turing's notion of anthropomorphic AI was extended by cultural praxis and techno-utopian imaginaries in the decades after his death.

While Kurzweil conceptualises his experiment in generative literature as a stepping stone in the journey towards anthropomorphic AI, Charles Hartman approaches generative literature as an end in itself, a new type of creative activity afforded by computer technology. Hartman chronicles his decades long experiments in generative literature in *Virtual Muse*, a short book published in 1996. The book is structured in chronological order: it begins with Hartman's first interaction with computers – learning the programming language FORTRAN IV and executing his programs via punched cards on a computer at his local college – and continues through to far more complex experiments with natural language processing. Throughout, Hartman outlines his numerous attempts at programming computers to participate in poetry-making.

Hartman's first experiment in generative literature was called RanLines (1996: 29). Hartman created a program that allowed the user to type twenty short lines of poetry and upload them to the computer. Then, each time the user pressed the return key, the computer would select a line at random and print it on the screen, creating a poem of randomly re-combined lines (1996: 29). While the machine Hartman was using had limited memory and processing power, and the premise of the program was simple, Hartman was pleased with RanLine's capacity to introduce a pseudo-random selection of the input text, which allowed for unlikely repetitions and juxtapositions of the lines (1996: 29). Hartman explains that because the computer has no sense of the words that it is randomly selecting, it can combine language with a type of disinterest of which the human poet is incapable (1996: 32). That is, while the human decision to use a certain word is contingent on its history and context – and therefore "can't be random" – the "advantage of the computer … lies in its perfect ignorance of language as such" so that it can select from a corpus in a way that is "completely arbitrary" (1996: 32, 100). In this sense, although a very simple program, for Hartman, RanLines signifies a collaboration between the human and the computer that best utilizes the different capacities of each actor. Indeed, for Hartman, poetry-making with RanLines was not a transference of poetic agency to the machine, but a dynamic interplay and negotiation between programmer, and the Sinclair ZX81 computer.

After RanLines, Hartman developed AutoPoet, a program which sought to establish "a common ground between what the computer… could realistically do and what I could plausibly define as writing poetry" (1996: 66). To make AutoPoet, Hartman compiled a machine-readable dictionary that included the number of syllables in each word, the stressed syllable, and the part of speech to which the word belonged (1996: 67). He then developed an n-gram generator so that AutoPoet could generate lines from the dictionary that statistically imitated English (1996: 67). Finally, Hartman incorporated an earlier program he had built that could

scan verse in order to filter out unmetrical lines from the generated text. The intention was that AutoPoet would be a "productive engine" that could generate the "poem's language on its own", without relying on human input beyond the original dictionary (1996: 66). With AutoPoet, Hartman moves away from his earlier collaborationist framework, and adopts an automative approach. As with Kurzweil, Hartman's objective with AutoPoet was to automate pre-existing forms of poetic production: that is, through the imitation of human language, Hartman wanted to make the computer generate "language on its own" (1996: 66). Despite achieving success in having the computer generate grammatical lines, Hartman was disappointed with the results of AutoPoet. Hartman proposes that the failure of the practice was caused by a misguided telos: in trying to have AutoPoet write poetry imitating existing poetic forms "on its own", he was conceptualising generative literature as a type of literary automation, with the computer imitating not only human language, but also notions of authorial autonomy and independence:

> As long as the goal was the imitation of a human poet – or as long as the poem's reader was encouraged to think that was the goal – I wasn't likely to get any farther. What's wrong with the AutoPoetry … is exactly that it's *imitation poetry*. All our habits of reading are called upon, all the old expectations, and then let down. (1996: 72)

Once Hartman realises that "the brute-force effort to make a machine into a human poet seems doomed" (1996: 4), he shifts his attention back to how the computer can be used for collaboration and discovery rather than imitation. This is evident in his final experiment, which makes use of his Prose program, a sentence generating algorithm that functioned by co-ordinating the contents of a dictionary and a generative grammar (1996: 73). Hartman utilised the Prose program for several different poetry-generating exercises. In one such instance, Hartman used Prose to generate paragraphs of random, yet grammatical sentences, on which he then provided commentary with footnotes (1996: 92–93). Together, the sentences and the commentary were taken as the poetic output. Again, the computer is used to generate and organise random sentences of language; however, then Hartman interprets the output of the machine, and incorporates this interpretation into the poetic text itself. In this sense, the collaborative construction of the text is foregrounded, including both the human and the technological actors in the output. The act of reading is conceptualised as a constitutive part of the poetic making, and part of the productive human-machine symbiosis. Hartman writes, "[the computer] was helping me think about poetry. Not simply confirming or codifying knowledge I already had … it was becoming a tool of discovery" (1996: 88).

In addition to conceptualising generative literature as a "tool of discover", in *Virtual Muse* Hartman also examines the post-human dimension of his generative literary practice. In fact, Hartman proposes that his practice is as much

about "the search for understanding between computers and people" as it is about generating poetic text (1996: 3). More specifically, through his practice, Hartman comes to realise that programming the computer to imitate human activity is limited to automating pre-existing human activities, and therefore configures the relationship between humans and machines as one of replacement, in which the computer poet supersedes the human at some point in the future: "a computer that becomes too autonomous begins to feel like a usurper" (1996: 36). For Hartman, like Licklider, a more suitable post-human figure for the digital paradigm resembles the cyborg: part human, part machine, an assemblage designed for exploration, and capable of extending human capacity through technological augmentation.

As can be seen from the above discussion, Hartman's hybrid digital poetics offers a point of comparison with Kurzweil's automative digital poetics: departing from a Turing-esque vision of imitation and replacement, it instantiates Lickliderian principles, in which the programmer collaborates with the computer to create hybrid assemblages of poetic agency that facilitate exploration of novel forms of poetry-making. One important point of difference between Licklider and Hartman, however, is that whereas Licklider talks about symbiosis, what Hartman appears to emphasise is collaboration and augmentation. That is, for Hartman, as creative practitioner, the human and machine don't fuse into one symbiotic entity, but rather, enter into a type of collaborative partnership in which the strengths of both are amplified. It is also important to emphasise that in *Virtual Muse*, Hartman is directly critical of programmers and poets who engage in the automative approach to generative literature. Specifically, issue is taken with the fact that the attempt to simulate human poetry with the computer does not productively transform poetic practice in any way, but simply re-inscribes pre-existing aesthetics and notions of individual authorship and readership into the sphere of generative literature. For example, as mentioned, Hartman rejects his own attempts at literary automation with AutoPoet, asserting that this approach is "doomed" (1996: 4). This critique of the automative approach is not only a rejection of a way of making poetry, but a denunciation of an associated Turing-like vision of anthropomorphic AI. That is, rejection of the "android" mode of generative literature is inextricably linked to a broader rejection of a Turing/Kurzweilian belief in the functional equivalence of humans and computational technologies.

In sum, in the microcosm of generative literature, we see how the visions of Turing and Licklider do not only inform cultural practice, but were also extended by them, while ramifying in broader debates about agency at the human-technology interface. In the final section of this paper, I return to the experiments of Karpathy and Matias in order to articulate how the visions, approaches and conflicts traced thus far extend into, and indeed contextualise and illuminate, contemporary approaches to and discourses about machine learning and AI.

Machine Shakespeare in Context

In 2010, J. Nathan Matias wrote a blog post about his poetry generator, SwiftSpeare. As explained earlier, this generator made use of machine learning software and a predictive text framework called TouchType, generally used for text messaging (Matias 2010: n. p.). In his blog post, Matias explains that he re-tooled this system to generate literature by training the algorithm on corpuses of poetry written by famous poets, including Shakespeare. Matias then assembled lines of poetry by choosing words purely from a list of next-word suggestions generated by the algorithm. In his blog post, Matias conceptualises the output poems as examples of "machine-learning-assisted poetry composition", a collaboration between a network of creators, including Shakespeare, the machine learning system, and himself (2010: n. p.).

Five years later, in 2015, Andrej Karpathy wrote a blog post about machine learning and text generation. In this post, Karpathy discusses various applications of Recurrent Neural Networks (RNN), a type of machine learning (2015: n. p.). One of the applications Karpathy outlines is "character-level language models", whereby one feeds the RNN a large corpus of text as training data, and "ask[s] it to model the probability distribution of the next character in the sequence of previous characters" (2010: n. p.). In other words, just as Matias' TouchType system was trained on a corpus of text messages to predict the messager's next word, the RNN is trained on a corpus of text, which it can then re-generate with probabilistic accuracy, although at the level of the character rather than the word. In one application of this technique, Karpathy trained an RNN on the complete works of Shakespeare, and then had it regenerate, one character at a time, a Shakespearean text. In his blog post, Karpathy writes that he "can barely recognize the samples [generated texts] from actual Shakespeare" (2010: n. p.). In other words, Karpathy conceptualises the RNN as a type of Shakespeare automation system, one that works well because it produces output indistinguishable from Shakespeare's literary works.

While Matias and Karpathy's experiments in generative literature utilise machine learning techniques developed in the last decade, they can be usefully contextualised within the historical framework I have outlined above. Specifically, Karpathy's blog post extends Turing's vision of machine intelligence and Kurzweil's corresponding automative approach to generating poetry. That is, the RNN system is trained on Shakespearian text, which it then imitates with a high degree of autonomy. The success of the system is measured according to whether it produces output that is indistinguishable from the original Shakespeare, like a Turing test. Furthermore, Karpathy makes the implication, by uploading a 100,000 character Shakespeare play, that RNNs are in some way an "unreasonably effective" automation of the creative process (2015: n. p.). Importantly, Karparthy's blog post does not limits itself to the application of RNNs in generating Shakespeare, but also shows their effectiveness in essay writing, algebraic geometry,

writing Linux source code, image recognition, and inductive reasoning (2015: n. p.). Indeed, Karpathy's blog post could well be taken as an extension, if not a vindication of, Turing's vision that one day computers "would enter any one of the fields normally covered by the human intellect, and eventually compete on equal terms" (Turing 2013: 358). It also extends Kurzweil's application of the Turing test as a methodology for assessing domain-specific intelligence.

By comparison, Matias' Swift-Speare experiment extends Licklider's vision of human-machine collaboration, and Hartman's corresponding approach to generative poetry. While he uses machine learning methods to generate text, he incorporates a human element into the system, amplifying the strengths of the computational and human actors in the creative act. As with both Licklider and Hartman, Matias' focus is less on automating poetic creation, as discovering new creative potentials that were previously impossible without the given computational tool. In an interview, he explains that Swift-Speare is not an attempt at replicating "existing poems" but searching for "probable poetry that has not yet been written" (Lomas 2014: n. p.).

However, it is important to note that while Matias does extend some Lickliderian principles into Swift-Speare, his practice and conception of AI is also influenced by the competing Turing/Kurzweil framework. That is, there is a convergence of automative and hybrid approaches in Matias experiment in generative literature; specifically, while he conceives of generating literature as a collaborative pursuit, the purpose is still to replicate Shakespeare with a degree of accuracy, rather than trying to find entirely novel forms of literary expression, like Hartman. Indeed, Matias suggests that the collaboration between human and machine is only a temporary state-of-play induced by a lack of technological prowess, and that at a point in the not too distant future, we will "see a successful automated poet" (Lomas 2014: n. p.). It is interesting that Matias uses the word "successful" to describe the automation of poetry, as if the collaborative approach is somehow a temporary failure on the journey to full technological potential. It is also telling that unlike Hartman and Licklider, Matias is not critical of Kurzweilian attempt to automate literature, nor the corresponding view of a progression towards technological perfection. It appears that for Matias, the collaborationist approach is more a result of the technological limitations of the present, than an ideological preference for human-machine symbiosis over android automation.

I propose that this is an indication of a broader capitulation to Turing/Kurzweil's vision for AI in our contemporary moment, and a diminishing understanding of the critical function alternative visions of AI, like Licklider's symbiotic intelligence, can offer. Increasingly, conversations about AI take on the tone and themes of Elon Musk-like predictions about a future in which intelligent robots replace humans first as workers, and then as a species. These conversations often overlook the ways in which technological developments afford new forms of collaborative practice between humans and machines. Briefly put, in the contemporary imaginary, AI is about the androids of the future, not the cyborgs of the

present. However, as I have shown, this Turing-derived vision of AI is not an inevitable future, but one option in amongst competing visions. Licklider's vision of augmented intelligence offers one such alternative, and is useful in that it offers strategies of critique for the automative obsession with imitation and replacement. Thus, the purpose of tracing a history from Turing and Licklider through Kurzweil and Hartman up to Karpathy and Matias has been to construct a historical perspective that tempers the default techno-utopian perspective that fetishizes new AI technologies, and in so doing open our eyes to alternative conceptions of machine intelligence. Doing so not only allows us to develop modes of critique in the present, but makes the future appear less technologically determined, more a matter of conflict and debate at the intersection of technology and culture.

References

Aarseth, Espen J. (1997): *Cybertext: perspectives on ergodic literature*. JHU Press.

Arulkumaran, Kai, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. (2017): "A Brief Survey of Deep Reinforcement Learning." (https://arxiv.org/pdf/1708.05866.pdf.) [To appear in *IEEE Signal Processing Magazine*]

Berlin, Isaiah (1960): *The age of enlightenment; the 18th century philosophers*. New American Library.

Bogost, Ian (2017): "Why Zuckerberg and Musk Are Fighting About the Robot Future." *The Atlantic*.

Crevier, Daniel (1996): *AI: the tumultuous history of the search for artificial intelligence*. New York, NY: Basic Books.

De, La Mettrie Julien Offray, and Paul-Laurent Assoun (1981): *LHomme machine*. Denoel/Gonthier.

Dennett, Daniel (2014 [2004]): "Can Machines Think." In: *Alan Turing: Life and Legacy of a Great Thinker*, edited by Christof Teuscher, 295–316.

Dick, Philip K (2008 [1968]): *Do Androids Dream of Electric Sheep?* Oxford: Oxford University Press.

Dreyfus, Hubert L. (1973): *What Computers Can't Do: a critique of artificial reason*. New York: Harper & Row.

Engelbart, Douglas (2001 [1962]): "Augmenting human intellect: a conceptual framework." In *PACKER, Randall and JORDAN, Ken. Multimedia. From Wagner to Virtual Reality.*, 64–90. WW Norton.

Floridi, Luciano (1999): *Philosophy and Computing: an introduction*. London: Routledge, 1999.

Funkhouser, Christopher Thompson (2007): *Prehistoric digital poetry: an archaeology of forms, 1959–1995*. The University of Alabama Press.

Haraway, Donna (2013 [1991]): *Simians, cyborgs, and women: The reinvention of nature*. Routledge.

Hartman, Charles O (1996): *Virtual Muse: Experiments in Computer Poetry*. University Press of New England.

Hayles, N. Katherine (1999): *How we became posthuman: Virtual bodies in cybernetics, literature, and informatics*. University of Chicago Press.

Karpathy, Andrej (2015): 'The Unreasonable Effectiveness of Recurrent Neural Networks.' *Andrej Karpathy blog*, May 21, 2015. (http://karpathy.github.io/)

Kurzweil, Ray (1999): "A (Kind of) Turing Test." Kurzweil CyberArt Technologies. (http://www.kurzweilcyberart.com/.)

Kurzweil, Ray (2000): *The age of spiritual machines: When computers exceed human intelligence*. Penguin.

Kurzweil, Ray (2000): "RKCP: How It Works." Kurzweil CyberArt Technologies. (http://www.kurzweilcyberart.com/)

Kurzweil, Ray (2001): "Ray Kurzweil's Cybernetic Poet." Kurzweil CyberArt Technologies. (http://www.kurzweilcyberart.com/)

Kurzweil, Ray (2012): "Locked in His Chinese Room." In *Are We Spiritual Machines? Ray Kurzweil vs. the Critics of Strong A. I.*, edited by Jay W. Richards, 128–70. Free Press, 2012.

Lem, Stanisław (1984): *Imaginary magnitude*. San Diego, CA: Harcourt Brace Jovanovich.

Licklider, J. C. R. (1960): "Man-Computer Symbiosis." *IRE Transactions on Human Factors in Electronics*HFE-1, no. 1: 4–11.

Lomas, Natasha (2014): "Read The Sonnet Co-Authored By Shakespeare, An MIT PhD Student & A Machine-Learning Algorithm." *Techcrunch*.

Marr, Bernard (2016): "A Short History of Machine Learning – Every Manager Should Read." *Forbes*.

Matias, J. Nathan (2010): "Swift-Speare: Statistical Poetry." *J. Nathan Matias Creative Portfolio* (web log), (http://natematias.com/)

McCorduck, Pamela, Marvin Minsky, Oliver G. Selfridge, and Herbert A. Simon (1977): 'History of Artificial Intelligence.' In *IJCAI*, pp. 951–954.

Mitchell, Tom M. (1997): *Machine learning*. New York: McGraw-Hill Book Company.

Newell, Allen, J. C. Shaw, and H. A. Simon (1983): "Empirical Explorations with the Logic Theory Machine: A Case Study in Heuristics." *Automation of Reasoning*, ACM Digital Library.

Newell, Allen, and Herbert A. Simon (1976): "Computer science as empirical inquiry: symbols and search." *Communications of the ACM* 19, no. 3: 113–26. ACM Digital Library.

Oppy, Graham and Dowe, David (2016): "The Turing Test", The Stanford Encyclopedia of Philosophy.

Papert, Seymour (1993): *Mindstorms: children, computers, and powerful ideas*. New York: Basicbooks.

Searle, John (2001 [1980]): "Chinese Room Argument, The." *Encyclopaedia of Cognitive Science*,

Shannon, Claude E. (1948): "A Mathematical Theory of Communication." Bell System Technical Journal 27.3: 379–423. Harvard Online.

Turing, Alan M. (1937): "On Computable Numbers, with an Application to the Entscheidungsproblem." Proceedings of the London Mathematical Society S2–42.: 230–65. Wiley Online Library.

Turing, Alan M. (1950): "Computing Machinery And Intelligence." *Mind*LIX, no. 236: 433–60.

Turing, Alan M. (2013): *The Essential Turing: seminal writings in computing, logic, philosophy, artificial intelligence, and artificial life "plus" the secrets of enigma*. Edited by B. Jack Copeland. Oxford: Oxford University Press.