Criação de narrativas através de um sistema multiagente

Jonath W. Herdt
jonathherdt@edu.univali.br
Universidade do Vale do Itajaí
Itajaí, Santa Catarina, BR

Rudimar L.S. Dazzi
rudimar@univali.br
Universidade do Vale do Itajaí
Itajaí, Santa Catarina, BR

Figura 1: Inteligência Artificial Narradora

ABSTRACT

This paper presents a system with agents capable of achieving objectives assigned to them by reading the environment in which they are inserted. A narrator capable of generating a textual narrative by observing the interactions and actions carried out by the agents was also developed , making it possible to create consistent and well-structured stories. To analyze the performance of the system, a survey was carried out. The levels of interest, coherence, predictability and creativity were evaluated on a scale from 1 to 5, which obtained average ratings of 3.10, 3.30, 2.26 and 3.56, respectively.

KEYWORDS

Natural language generation, Multi-agent systems, Intelligent Agents

1 INTRODUÇÃO

Narrativa como entretenimento, na forma de histórias orais, escritas ou visuais, desempenha um papel central nas vidas humanas sociais e de lazer. Existem evidências que sugerem que seres humanos, constroem estruturas cognitivas que representam os eventos reais de suas vidas usando modelos similares àqueles usados para narrativas com o intuito de entender melhor o mundo a sua volta. [2] Essa inteligência narrativa [6] é uma parte central do processo cognitivo que seres humanos empregam em uma gama de experiências, desde conteúdo de entretenimento à aprendizado ativo. A habilidade de gerar narrativas é a capacidade de criar e estruturar uma sequência de eventos que possam ser compreendidos como elemento de uma história. Esta habilidade é de extrema importância para sistemas que desejam utilizar histórias para entretenimento, treinamento ou educação. [12]

Caso o narrador seja um sistema com Inteligência Artificial (IA), precisará simular, de modo efetivo, um mundo coerente dentro da história e controlar um número relativo de personagens, tornando-a coerente de um ponto de vista humano. Assim, para criar histórias coerentes, seja interativamente ou não, esse sistema precisa de métodos de diversos subcampos de IA, como agentes virtuais e sistemas multiagentes, raciocínio sobre crenças e intenções, planejamento multiagente, computação afetiva, e sistemas de diálogo. [1]

Portanto, este trabalho busca identificar e compreender quais os elementos de enredo, narração e apresentação são usados para construir uma boa história. Em sequência, tais elementos são utilizados para a implementação de um sistema multiagente. Estes agentes podem ser Atores ou Narrador. Os atores assumem papéis de personagens dentro da narrativa, tomando decisões e buscando alcançar seus próprios objetivos. O Narrador deve observar todas as ações tomadas pelos atores e descreve-las em formato de texto utilizando um módulo de geração de linguagem natural. Deste modo, o sistema tem o objetivo de criar uma história consistente, com uma sequência de eventos que pareça natural e de acordo com o mundo em que está inserida.

2 TRABALHOS RELACIONADOS

[3] propõem uma arquitetura de narrativa baseada em agentes, que combina autoria de histórias baseada em enredo e modelagem por comportamento de personagem, chamada de DIRACT. DIRACT é uma abreviação para “Dirige e Atua” e representa as duas principais funções de uma narrativa: estrutura de história e presença de história;

[1] propôs a utilização de uma linguagem de planejamento multiagente (MAPL – Multiagent Planning Language). Essa abordagem

traz alguns benefícios citados pelo autor, como: a) Conhecimento incompleto, b) Crenças e crenças mútuas entre personagens, c) Ações de busca de conhecimento, d)*Initial State Revision*;

[16] fizeram um estudo de geração de histórias em linguagem natural a partir de qualquer título (tópico) dado, propondo decompor geração de histórias em dois passos: 1) planejamento de história que produz um esboço e 2) realização de superfície que compõe texto em linguagem natural baseado no esboço; e

[10] apresentam um modelo de *transformer* chamado PlotMachine. Esse modelo aprende a transformar um esboço em uma história de vários parágrafos, usando blocos dinâmicos de memória que mantém os estados implícitos do enredo computado usando o esboço e a história gerada até então.

3 SISTEMAS MULTIAGENTES

Um agente é uma entidade de software que exibe um comportamento autônomo. Há um consenso de que a autonomia é a ideia central da noção de agência [15]. A autonomia é a capacidade do agente de agir por seus próprios objetivos, sem a intervenção de outrem. Um agente também deve ser proativo e orientado a objetivos, o qual está situado em algum ambiente em que é capaz de realizar ações para alcançar seus próprios objetivos e a partir do qual percebe alterações [13].

Os sistemas multiagentes são sistemas abertos, em que agentes não são necessariamente projetados para atingir um objetivo comum, podendo ingressar e sair do sistema de maneira dinâmica. Neste caso, a chegada dinâmica de agentes desconhecidos precisa ser levada em consideração, bem como a possível existência de comportamento não benevolente no curso das interações [17]. Neste tipo de sistema, um conjunto de agentes autônomos interagem com o objetivo de alcançar a resolução de um problema que está além das capacidades de apenas um indivíduo [8].

3.1 Programação Orientada a Agentes

Este trabalho faz uso da linguagem de programação orientada a agentes chamada "GOAL", que é baseada em regras para a programação de agentes cognitivos que interagem com o ambiente e entre eles. A linguagem GOAL promove programação com estados cognitivos, esses estados têm estruturas adicionais se comparados com base de dados mais tradicionais, e são muito diferentes de estados na maioria das outras linguagens de programação. Agentes cognitivos são agentes autômatos para tomada de decisão, que derivam suas escolhas de ação de suas crenças e objetivos [5].

Agentes controlam entidades em ambientes e decidem o que fazer com essas entidades. O centro da programação orientada a agentes é o modelo de tomada de decisão, onde o agente faz decisões baseado em suas crenças e objetivos. Essa habilidade de tomar decisões baseado em tais elementos, dá o nome à agentes cognitivos. Ambientes podem ser praticamente qualquer coisa variando de mundos de brinquedos, simuladores, jogos, mundos virtuais até robôs físicos [5].

Um agente cognitivo possui três habilidades centrais que permitem efetivamente controle e interação efetivamente com uma entidade em um ambiente. A primeira habilidade é processamento de evento, que permite o agente processar eventos como percepções que recebe do ambiente, e mensagens que outros agentes trocam entre si, para atualizar suas crenças. Eventos também podem fazer com que o agente atualize seu objetivo. A segunda habilidade é a representação de conhecimento, que permite ao agente manter um modelo do ambiente, que é essencial para manter controle do estado do ambiente. Isso também permite o agente raciocinar sobre o ambiente e representar o que o agente busca alcançar. A terceira habilidade é a tomada de decisão, que possibilita o agente a decidir qual próximo passo tomar e selecionar uma ação. Essas habilidades centrais estão relacionadas à importantes áreas da inteligência artificial [5].

4 GERAÇÃO DE LINGUAGEM NATURAL

Geração de linguagem natural é um subcampo de inteligência artificial e linguística computacional que foca em sistemas computacionais que podem produzir textos em linguagem humana. Tipicamente começando com uma representação não linguística da informação como entrada, sistemas Geração de Linguagem Natural (NLG - Natural Language Generation) usam conhecimento sobre a linguagem e o domínio da aplicação para automaticamente produzir documentos, relatórios, explicação, mensagens de ajuda, e outros tipos de textos [11].

4.1 Generative Pre-Training

Antes da publicação de [9], maior parte dos modelos NLP de estado da arte eram treinados especificamente em uma tarefa, como classificação semântica, vinculação textual, etc. utilizando aprendizado supervisionado. Porém, modelos supervisionados tem duas grandes limitações: (i) eles precisam de uma grande quantidade de dados anotados para aprender uma tarefa particular o que geralmente é difícil de encontrar, (ii) eles falham em generalizar para tarefas diferentes do que eles foram treinados para fazer. O Pré-Treinamento Generativo (GPT - Generative Pre-Training) propõe aprendizado de um modelo de linguagem generativo, utilizando dados não rotulados e então ajuste fino do modelo fornecendo exemplos de tarefas específicas como classificação, análise de semântica, vinculação textual, etc [14].

Para este trabalho, foi utilizado mais especificamente o modelo de linguagem GPT-2, que possui 1,5 bilhão de parâmetros, 10 vezes mais que GPT (117 milhões de parâmetros). Utilizando 48 camadas para decodificador, estrutura de transformer com auto atenção mascarada para treinar o modelo de linguagem. O mascaramento ajudou o modelo de linguagem atingir o objetivo desse modelo, mesmo que ele não tinha acesso às palavras subsequentes à direita da palavra atual. Um vocabulário de 50257 tokens foi utilizado, juntamente com um batch de 512 de tamanho com uma janela de contexto de 1024 tokens [9].

4.2 Preenchendo Lacunas com Modelos de Linguagem

Preenchimento de lacunas é uma tarefa de prever palavras que encaixem em espaços em branco dentro de um texto, que sejam consistentes com o texto anterior e posterior à lacuna. Sistemas capazes de preencher lacunas tem o potencial de enriquecer aplicações que tem como tarefa auxiliar humanos em edição ou revisão de texto, conectando fragmentos de ideias e restaurando documentos antigos [4].

Preenchimento por Modelagem de Linguagem (ILM - Infilling by Languagem Modeling) é um framework que possibilita Modelos de Linguagem (LMs - Language Models) a preencher espaços em branco de tamanhos variáveis, enquanto preservando qualidade de geração, amostragem eficiente e simplicidade conceitual. Este framework envolve uma formulação direta na tarefa de preenchimento, que pode ser aprendida efetivamente por qualquer arquitetura de LM. Como mostrado na Figura 2, o framework concatena texto mascarado artificialmente com o texto original, e adota um procedimento padrão para treinamento (ou ajuste fino) do LM. Uma vez treinado, o preenchimento pode ser feito em um documento com espaços em branco, utilizando o LM para gerar o texto e então substituir os espaços em branco com o texto gerado [4].

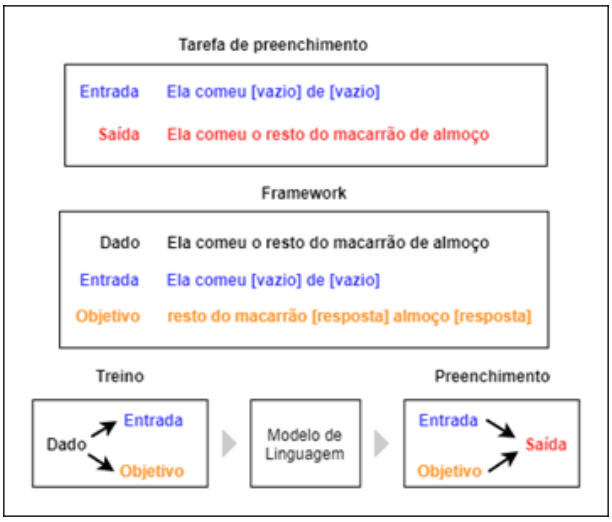

Figura 2: Funcionamento do framework de preenchimento de lacuna

5 DESENVOLVIMENTO

A Figura 3 ilustra o sistema, onde estão contidas as duas entidades principais deste trabalho, os Atores e o Narrador. Para atingir o objetivo de gerar narrativas, este trabalho fez uso da linguagem de programação orientada a agentes GOAL, do framework de preenchimento de lacunas em texto ILM e do modelo de linguagem GPT-2.

5.1 Atores

Os agentes Atores são responsáveis por interpretarem os personagens dentro da narrativa, cada um deles possuindo um nome de personagem, base de conhecimento e objetivos próprios, para que desta forma possam diferenciar-se uns dos outros e criar variedade de ações dentro da narrativa. Um objetivo diferente a alcançar para cada ator dá fluidez para a narrativa, além de a personalidade de cada um ser diferenciada por suas bases de conhecimento. O código-fonte dos cenários e atores utilizados no trabalho estão disponíveis em um repositório do GitHub: https://github.com/JonathWesley/GoalAgentsExamples.

A modelagem de um agente Ator é ilustrada na Figura 4, onde cada Ator está ligado a uma entidade dentro do ambiente e através

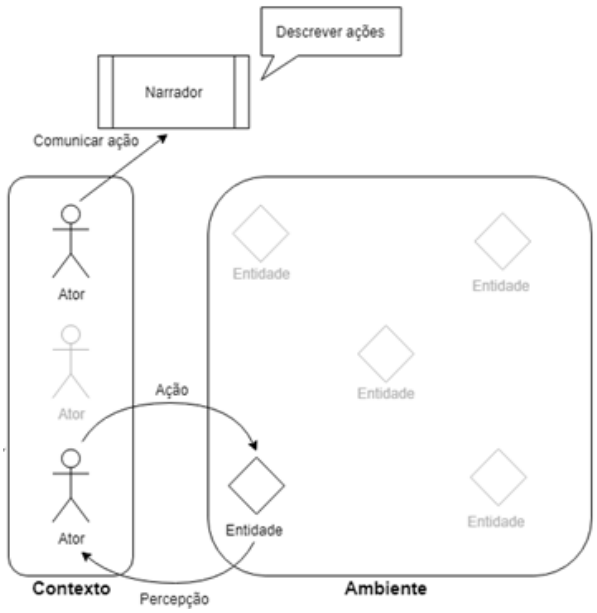

Figura 3: Exemplificação do funcionamento do sistema

dessa entidade o agente realiza a percepção desse ambiente, recebendo informações do que a entidade pode perceber. A percepção do agente altera suas crenças, sendo essas crenças a representação do estado do ambiente para o agente. Esse processo é importante para que o agente tenha noção apenas da parte do ambiente em que a entidade se encontra, dando assim consistência as futuras ações a serem tomadas, pois serão baseadas apenas nas crenças atuais do agente, podendo gerar conflitos interessantes a partir das mudanças no ambiente que não são percebidas pelo agente.

Além das crenças, outra parte importante para a tomada de decisão do agente Ator são seus conhecimentos, estes diferentemente das crenças, são imutáveis. Portanto, no contexto de uma narrativa e representação de personagem, os conhecimentos de um agente podem ser utilizados para representar a personalidade do mesmo. Um Ator possui um ou mais objetivos que deve alcançar. Estes objetivos são pequenos, como pegar uma maçã ou atravessar o quarto, mas com a junção de vários objetivos seguidos pode-se dirigir o personagem para encontros interessantes. A importância destes objetivos foi justamente criar interações conflituosas entre dois ou mais personagens na narrativa, tirando proveito da narrativa gerada com isto. Por último, um Ator possui uma lista de ações que pode tomar. Cada uma dessas ações possui uma ou mais pré-condições para que ela possa ser executada, e uma ou mais pós-condições que são realizadas ao final da execução da ação especificada.

Como saída dos Atores dentro do sistema, ocorreu a comunicação de cada ação tomada para o narrador, de uma forma padronizada para que o narrador pudesse acrescentar complexidade nas frases que compuseram a narrativa. O padrão da mensagem foi uma frase contendo a seguinte sequência de informações: (i) Nome do Ator com um token [vazio] ao final; (ii) nome da ação tomada com um token [vazio] ao final; (iii) opcionalmente, pode existir o nome de uma entidade secundária (objeto) com o que o Ator interagiu com

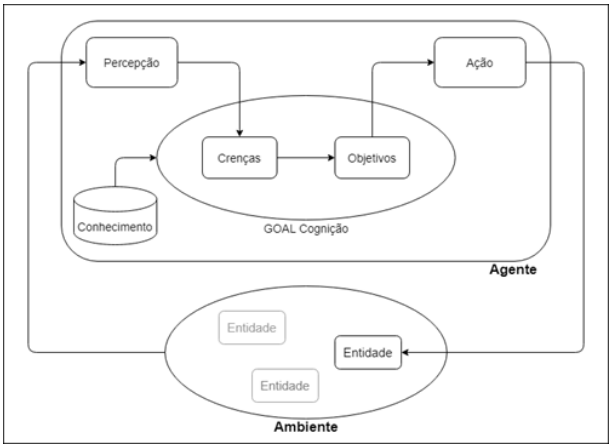

Figura 4: Modelagem do Ator

um token [vazio] ao final; e por último, (iv) também opcionalmente, o nome de outro Ator com quem interagiu. A Figura 5 demonstra esse padrão de comunicação, e apresenta alguns exemplos de mensagens.

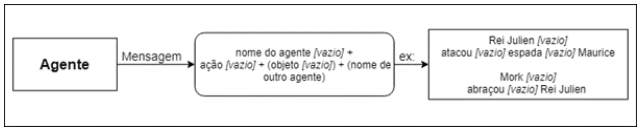

Figura 5: Saída através de mensagens do Ator

5.2 Narrador

O narrador foi responsável pela transformação das mensagens enviadas pelos autores em frases fluídas e complexas, criando assim uma narrativa textual concisa e interessante. Para tal, o narrador foi implementado da maneira que a Figura 6 apresenta. Uma vez que a mensagem de um Ator foi recebida, ela foi dada como entrada para o modelo de linguagem GPT-2 previamente treinado para o preenchimento de tokens [vazio]. O módulo de GPT-2 processa então a frase e tem como saída a(s) string(s) de [resposta], que são denominados como objetivo dentro da entidade. Com o objetivo formado, o Narrador utiliza a frase de entrada e substitui os tokens [vazio] pelas strings de [resposta], preenchendo respectivamente cada lacuna e formando a frase de saída.

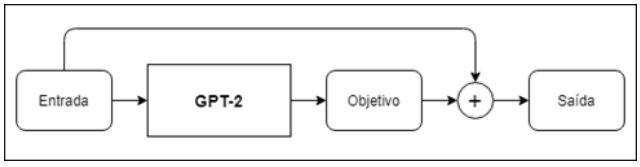

Figura 6: Modelagem do Narrador

Com o framework de preenchimento de lacunas ILM, foi possível treinar o modelo de linguagem GPT-2 para que ele realizasse o preenchimento de lacunas em textos. Neste trabalho, foi justamente esse preenchimento que trouxe complexidade para as frases dentro da narrativa. Foi utilizado o conjunto de dados disponibilizado por [7], que continham 100 mil exemplos de histórias curtas, sendo estas histórias feitas de um título e cinco sentenças cada.

O Narrador teve seu desenvolvimento feito utilizando a linguagem Python, o que acabou deixando um pouco menos custoso seu desenvolvimento, já que Python é uma linguagem extensivamente utilizada. Seguindo o repositório do GitHub: https://github.com/JonathWesley/storyteller, que contém todo o código-fonte do Narrador, o primeiro passo tomado pelo algoritmo é carregar o modelo de linguagem GPT-2 treinado. Em seguida o algoritmo carrega alguns tokens adicionais para melhorar na geração de cada frase, fornecendo um vocabulário maior para o Narrador. No próximo passo, é utilizada a biblioteca PyTorch juntamente com a Transformers, para definir o ambiente de execução do modelo treinado, e disponibilizar a CPU da máquina para a execução do mesmo. Então é carregado o arquivo de entrada, que é lido linha por linha, fazendo o preenchimento dos tokens [vazio] uma frase por vez. E por fim é salvo o resultado, com todas as frases preenchidas, em um arquivo de texto.

Com a frase passando pelo módulo de GPT-2, foi gerado o que é chamado de objetivo, uma string contendo o conteúdo que substitui cada uma das lacunas, separados por um token [resposta]. Com o objetivo em mãos, o narrador pôde utilizar a entrada, substituindo cada uma das lacunas por seus respectivos preenchimentos, desta forma a frase de saída do narrador foi formada. Um exemplo dessa execução é ilustrado na Figura 7.

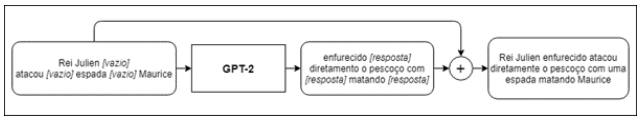

Figura 7: Saída do Narrador

6 RESULTADOS

Para analisar o desempenho do sistema de criação de narrativas desenvolvido, realizou-se uma pesquisa, através da plataforma Google Forms, formulário disponibilizado no seguinte link: https://forms.gle/ygYZgWZfRaVApgh99. Nesta pesquisa os participantes fizeram a leitura de 3 histórias com cenários diferentes para avaliá-los quanto ao nível de interesse, coerência e previsibilidade. Ao final, foram apresentadas duas histórias geradas do mesmo cenário, que foram avaliadas pelos participantes para mensurar o quão criativo o sistema poderia ser quando comparadas as duas histórias. A Figura 8 apresenta uma das histórias disponibilizadas nessa pesquisa.

O questionário foi respondido ao todo por 25 pessoas, possibilitando a avaliação por meio das médias das notas. O primeiro quesito em que as 3 histórias de exemplo foram avaliadas foi o quão interessante elas são, para isso cada participante deu uma nota de 1 a 5, onde 5 significava muito interessante, e 1 significava pouco interessante, a Figura 9 demonstra os resultados desta primeira avaliação. O segundo quesito a ser avaliado foi a coerência das histórias, da mesma forma cada participante deu uma nota equivalente de 1 a 5, a Figura 10 mostra os resultados desta segunda

Storage
Ramon works on construction warehouse.
Ramon enjoys working at a warehouse.
He works from the trees at the bottom.
The wind often blows in at night.
At night he sometimes makes music.
Ramon moves his boxes .
He visits his hotel to rest.
Ramon heard about his boss said wants a box.
Ramon started by moving boxes.
He thought it would be difficult to do so.
Ramon moved his box one to the table and almost never dropped anything.
Ramon moved the box two to the table and never dropped anything.
Ramon moved another box four to the box three.
Ramon moved a box five to the box two exhausted, he checked and found the box.
Ramon moved the box one to box five.
It was easier to move box one than moving box five.
Ramon moved the box six to the box four.
The next morning he dropped the box six.
Ramon never had finished working, the boxes had been moved again.
The old box now are the size of 10 and the same shape.

Figura 8: Exemplo de história disponibilizada na pesquisa

avaliação. O terceiro quesito a ser avaliado foi a previsibilidade das histórias, porém é preciso tomar atenção neste caso em específico, pois diferentemente dos outros dois quesitos, para uma história o quanto menos previsível ela for melhor. Logo, uma nota 1 seria um resultado melhor do que uma nota 5, a Figura 11 apresenta os resultados desta terceira avaliação. O último quesito a ser avaliado foi a criatividade do sistema em geral, para isso foram apresentadas duas histórias geradas a partir do mesmo cenário, e então os participantes deram uma nota pela criatividade do sistema em criar histórias diferentes, a Figura 12 demonstra os resultados desta última avaliação.

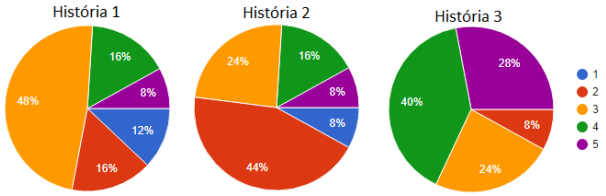

Figura 9: Notas de interesse das histórias

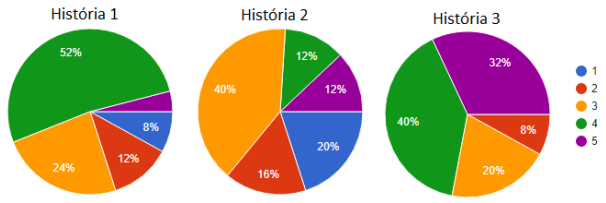

Figura 10: Notas de coerência das histórias

Os resultados desta pesquisa foram utilizados para calcular a média geral de cada um dos quesitos de avaliação. O cálculo dessa média se dá por aritmética simples, já que os valores possuem a mesma importância (peso). Com este cálculo, observou-se que o

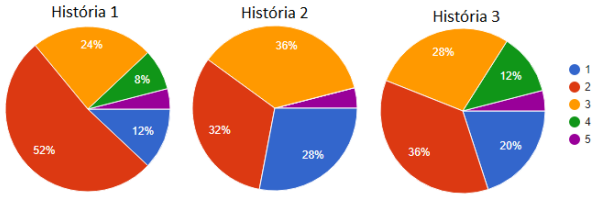

Figura 11: Notas de previsibilidade das histórias

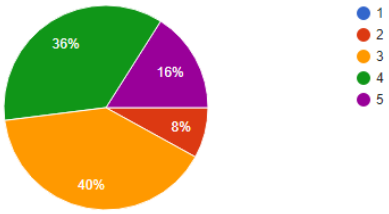

Figura 12: Notas de criatividade das histórias

interesse, a coerência, a previsibilidade e a criatividade geral das histórias obtiveram uma média de classificação de 3,17, 3,36, 2,34 e 3,6, respectivamente.

Para Rashkin et al (2020) e Yao et al (2019) pode-se notar quais atributos se destacam nas histórias resultantes. No caso de Yao et al (2019), o atributo que se sobressaiu foi a coerência das histórias, enquanto no caso de Rashkin et al (2020), o interesse teve destaque. Em relação a este trabalho, ambos interesse e coerência tiveram as menores notas perante aos requisitos de coerência e interesse levantados.

Para o caso de Cai et al (2010) e Brenner (2010), onde ambos utilizam de sistemas multiagentes, pode-se fazer comparações com a capacidade de interação desses agentes. Onde o trabalho de Cai et al (2010) possui múltiplos atores em suas narrativas, porém estes atores possuem poucos ciclos de interação, o que deixa a narrativa relativamente curta demais, e consequentemente menos interessante. Enquanto o trabalho de Brenner (2010), acaba sendo o mais parecido com este trabalho, pois leva a narrativa com uma duração maior. Perante os quesitos de interesse e coerência, o trabalho de Brenner acabou recebendo uma avaliação melhor.

7 CONCLUSÃO

Neste trabalho, foi desenvolvido um projeto de sistema multiagente, cujo objetivo foi a criação de narrativas curtas que sejam coerentes e interessantes. Foram realizadas as implementações dos agentes na ferramenta Eclipse com plugin para a linguagem GOAL, tendo em mente todas as funcionalidades dos mesmos. Em seguida, foi desenvolvido o narrador, utilizando ILM para treinar o modelo GPT-2 e Python para o restante da implementação. Com isto, foram integradas as duas implementações, para realizar os testes e analisar os resultados obtidos.

O trabalho apresentou resultados significativos, e de forma geral pode-se considerar o objetivo geral cumprido uma vez que o sistema consegue criar narrativas a partir de seus agentes. Os resultados, por sua vez, se mostraram satisfatórios, mesmo havendo espaço para

melhorias, principalmente em questões de interesse e coerência das histórias.

REFERENCES

[1] M Brenner. 2010. *Creating Dynamic Story Plots with Continual Multiagent Planning*.
[2] J. Bruner. 1990. Acts of Meaning. *MA: Harvard University Press* (1990).
[3] Y. Cai, Z. Shen, C. Miao, and A. Tan. 2010. DIRACT: Agent-based Interactive Storytelling. *International Conference on Web Intelligence and Intelligent Agent Technology* (2010).
[4] C. Donahue, M. Lee, and P. Liang. 2020. *Enabling Language Models to Fill in the Blanks*. 2492–2501.
[5] K.V. Hindriks. 2018. Programming Cognitive Agents in Goal. (2018).
[6] M. Mateas and P. Sengers. 2003. Narrative Intelligence. *Philadelphia: John Benjamins* (2003).
[7] N. Mostafazadeh, N. Chambers, X. He, D. Parikh, D. Batra, L. Vanderwende, P. Kohli, and J. Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. *North American Association for Computational Linguistics (NAACL)* (2016).
[8] Greg O'Hare and Nicholas R. Jennings. 2010. *Foundations of distributed artificial intelligence*. John Wiley and Sons.
[9] A. Radford, K. Narasimhan, T. Salimans, and Sutskever I. 2018. Improving language understanding by generative pre-training. (2018).
[10] H. Rashkin, A. Celikyilmaz, Y. Choi, and Gao J. 2020. PLOTMACHINES: Outline-Conditioned Generation with Dynamic Plot State Tracking. (2020).
[11] E. Reiter and R. Dale. 2003. *Building Natural Language Generation Systems*. 1–22.
[12] M.O. Riedl and R.M. Young. 2004. An Intent-Driven Planner for Multi-Agent Story Generation. *Autonomous Agents and Multiagents Systems* (2004).
[13] S. J. Russel and P. Norvig. 2010. *Artificial intelligence. A modern approach* (3rd ed.). NJ: Prentice-Hall.
[14] Priya Shree. 2020. *The Journey of Open AI GPT models*. https://medium.com/walmartglobaltech/the-journey-of-open-ai-gpt-models-32d95b7b7fb2
[15] M. Wooldridge. 1999. *Intelligent Agents*. MIT Press.
[16] L. Yao, N. Peng, R. Weischedel, K. Knight, D. Zhao, and R. Yan. 2019. *Plan-and-Write: Towards Better Automatic Storytelling*.
[17] F. Zambonelli, N. R. Jennings, and M. Wooldridge. 2000. *Organisational Abstractions for the Analysis and Design of Multi-agent Systems*. 127–141.