

Utilizando *Question Answering* no Auxílio ao Processo de Ensino e Aprendizagem de Programação: Um Estudo de Caso com BERT e ChatGPT

Marcelo de L. Freire¹, Robson G. Fechine Feitosa^{1,5}, Yuri D. Santos²,
Hanna Menezes³, Guilherme Á. R. M. Esmeraldo¹, Harley M. de Mello¹,
Esdras L. Bispo Jr.⁴, Gustavo A. L. de Campos⁵

¹Instituto Federal de Educação, Ciência e Tecnologia do Ceará – Campus Crato (IFCE)
Rodovia CE 292, KM 15, Gisélia Pinheiro – CEP 63115-500 – Crato – CE – Brazil.

²Department of Theoretical Philosophy –
University of Groningen, 9712 GL – The Netherlands.

³Unidade Acadêmica de Ciência da Computação –
Universidade Federal de Campina Grande – PB – Brasil.

⁴Instituto de Ciências Exatas e Tecnológicas (ICET)
Universidade Federal de Jataí (UFJ), Goiás, Brasil

⁵Departamento de Ciências da Computação –
Universidade Estadual do Ceará – CE – Brasil.

marcelo.lima.freire95@aluno.ifce.edu.br,
{robsonfeitosa, guilhermealvaro, harleymello}@ifce.edu.br,
yuridavidsantos@gmail.com, hanna@copin.ufcg.edu.br,
bispojrr@ufj.edu.br, gustavo.campos@uece.br

Abstract. *Natural Language Processing is an area of Artificial Intelligence that has brought benefits to the most diverse processes of human activity. In this context, this work uses a Question Answering (QA) approach to help in the process of teaching and learning programming. For this purpose, it analyzed two case studies with different QA models and a database with 87 questions and answers related to teaching programming. Thus, the models achieved 62% and 85% accuracy in the first and second scenarios, respectively. The present work also discusses (i) some limitations of the approach, (ii) some qualitative pedagogical results, and, finally, indicates (iii) recommendations for future work.*

1. Introdução

No contexto da Inteligência Artificial (IA), o Processamento de Linguagem Natural (PLN) é uma área de pesquisa que vem sendo aplicada nos mais diversos setores das atividades humanas, por exemplo, desde a utilização de *chatbots* e assistentes virtuais, para auxiliar no processo de comunicação, até mesmo na tradução de texto; e, segundo [Jurafsky e Martin 2023], ele tem como objetivo permitir aos computadores a compreensão da linguagem natural humana, para a realização de diversas tarefas. Dentre as abordagens de PLN, o *Question Answering* (QA) é uma abordagem promissora no contexto de recuperação de informação. Ele pode ser aplicado no processo de ensino e aprendizagem

[Swathilakshmi et al. 2021, Farias 2019] em uma perspectiva geral, viabilizando um ambiente de consultas direcionadas a um domínio específico, inclusive como uma Tecnologia na Educação em Computação [Bispo Jr. et al. 2020]. Em [Swathilakshmi et al. 2021], por exemplo, destaca-se a oportunidade de aplicação de QA em fóruns de discussão, pois, à medida em que crescem em volume de perguntas, torna-se conveniente o uso de QA, conseqüentemente, dispensando a necessidade de filtros ou mesmo encaminhamentos por parte do especialista.

Diante desse contexto, o presente trabalho se utiliza de uma abordagem baseada em PLN, a partir de um ambiente de QA e uma base de conteúdos sobre programação de computadores, com o intuito de verificar se tal ambiente pode auxiliar o processo de ensino e aprendizagem de programação. Tal abordagem oferece a possibilidade de alunos realizarem consultas em linguagem natural sobre conteúdos diretamente alinhados com o tema de programação de computadores. Para isso, o trabalho foi organizado da seguinte forma: na Seção 2, são apresentados conceitos fundamentais ao tema de PLN e QA; na Seção 3, são descritos os passos metodológicos aplicados para o desenvolvimento do presente trabalho; na Seção 4, são apresentados e analisados os resultados alcançados; na Seção 5, são descritas algumas conclusões e indicações de trabalhos futuros.

2. Fundamentos

De acordo com [Lende e Raghuwanshi 2016], QA corresponde a um sistema no qual consultas são inseridas como entrada em linguagem natural, e permitem um processo de busca em documentos para extrair possíveis respostas para a referida consulta. Ou seja, trata-se de uma técnica que combina Recuperação de Informação, do inglês *Information-Retrieval* (IR), com PLN, tendo por objetivo responder, de forma automática, às perguntas elaboradas por humanos em uma linguagem natural. Dessa forma, os sistemas de QA vêm sendo utilizados em uma variedade de campos de pesquisa pela capacidade de tornar as informações mais acessíveis, por meio do raciocínio e da inferência sobre uma quantidade significativa de dados, disponíveis em uma linguagem natural [Soares e Parreiras 2020].

Segundo [Lende e Raghuwanshi 2016] as duas principais abordagens de QA são: *Open Domain* e *Closed Domain*. *Open Domain* preocupa-se com perguntas que são relevantes para todos os domínios. Nessa modalidade, a quantidade de dados que estão disponíveis é mais ampla e, a partir desses dados, o sistema extrai a resposta. Ele pode responder a qualquer pergunta sobre qualquer domínio, porém com um nível de precisão menor, dada a falta de especificidade do domínio [Lende e Raghuwanshi 2016]. Por outro lado, *Closed Domain* refere-se a perguntas específicas de um domínio e pode ser considerado como um processo mais simples, porque os sistemas de PLN podem fornecer conhecimento de domínio específico. Esse modelo disponibiliza um alto nível de precisão, mas é limitado a um único domínio [Lende e Raghuwanshi 2016].

Em [Roberts et al. 2020], é realizada uma análise do desempenho de modelos envolvendo QA, partindo da hipótese de que quanto maior o número de parâmetros do modelo, ele tende a reter mais informações. Os autores evidenciam que o desempenho de *Transfer Learning* (um método para reutilizar um modelo treinado para uma tarefa, ajustando-o, ou seja, aplicando o processo de *fine-tuning* para outra tarefa, mas dentro do mesmo contexto), em tarefas diversificadas, melhora à medida que o tamanho do modelo e a quantidade de pré-treinamento não supervisionado aumenta. Além disso, a empresa

OpenAI, disponibilizou acesso¹ ao seu modelo GPT (*Generative Pre-trained Transformer*), que se trata de um grande modelo QA de linguagem baseado em transformador com 175 bilhões de parâmetros, treinado a partir de um grande conjunto de dados oriundos de fontes como: *CommonCrawl*, *WebText dataset*, dois livros de *corpora* baseados na internet e Wikipedia em inglês [Brown et al. 2020].

3. Materiais e Métodos

O presente trabalho empregou um processo metodológico exploratório, que se iniciou com uma investigação bibliográfica seguida da aplicação de experimentos por meio de estudos de caso, o que, de acordo com [Gil et al. 2017], representa uma abordagem de pesquisa amplamente utilizada a qual compreende o estudo minucioso e exaustivo de um ou poucos casos, de forma a possibilitar seu conhecimento em profundidade. Para isso, foram delimitados dois cenários como estudos de caso: o primeiro se utiliza da abordagem proposta por [Farias 2019], através de um modelo de *Question Answering* na modalidade *Closed Domain* aplicado ao ensino de programação; o segundo, analisa o desempenho do modelo GPT, sob a mesma base de dados de perguntas adotada no primeiro cenário.

3.1. Materiais e Métodos para o Primeiro Cenário

Para a construção de um modelo de QA do primeiro cenário, foi necessário realizar as seguintes etapas: (i) seleção e análise da base de dados; (ii) extração das informações de interesse da base selecionada; (iii) adaptação do modelo para novo *corpus*; e, (iv) teste e validação do modelo.

Na etapa de seleção da base de dados, foram definidos os seguintes critérios: (a) conjunto de dados disponível publicamente para utilização no presente trabalho; (b) contexto educacional, ou seja, uma base de dados que pudesse ser aplicada em tarefas voltadas para o ensino-aprendizado de conteúdo de programação; além de (c) já ter sido utilizada em outras pesquisas que abordam o tema QA. A partir das especificações mencionadas, foi selecionado o *corpus*² disponibilizado por [Mohler et al. 2011], que realizou uma análise de desempenho de estudantes, com o propósito de mensurar a precisão de *short answering* (respostas resumidas) desenvolvidas por esses alunos.

O *corpus* escolhido dispõe de 87 questões no seguinte formato: perguntas e respostas elaboradas por docentes; identificador; respostas e notas referentes aos estudantes. Para tanto, no presente trabalho, realizou-se um pré-processamento dos dados, no qual optou-se por utilizar apenas perguntas e respostas elaboradas pelo professor, que corresponde à etapa (ii) do método aqui adotado para construção de um modelo de QA. Então, a partir da definição dos dados, foi estruturado um *dataset* com duas colunas (*title* e *paragraphs*) sendo a coluna *title* referente às questões (perguntas) e a coluna *paragraphs* referente às respostas elaboradas pelo professor.

Após a seleção do *corpus*, deu-se início aos ajustes da abordagem proposta por [Farias 2019], para adaptação do modelo conforme etapa (iii). Para isso, em [Farias 2019], utilizou-se duas estratégias, em que na primeira foi realizado um pré-processamento dos dados a partir do uso do TF-IDF (*Term Frequency - Inverse Document*

¹Disponível em: <https://chat.openai.com/chat>. Acesso em: 17 de março de 2023.

²Disponível em: http://web.eecs.umich.edu/~mihalcea/downloads/ShortAnswerGrading_v2.0.zip. Acesso em: 04 de abr. de 2023.

Frequency), sendo o resultado passado como entrada para o BERT (*Bidirectional Encoder Representations from Transformers*) pré-treinado com o *Stanford Question Answering Dataset (SQuAD)* [Rajpurkar et al. 2016]. O SQuAD utilizado, em sua versão 1.1, compreende um conjunto de dados contando mais de 100.000 perguntas elaboradas por *crowdworkers*³ sobre artigos da Wikipédia, sendo a resposta para cada pergunta uma passagem de texto de uma leitura relevante. Então nesta etapa, foi adicionado o *dataset* selecionado e ajustado conforme as etapas anteriores (i) e (ii). A segunda estratégia adotada por [Farias 2019] se utiliza de uma técnica de transferência de aprendizagem e *fine-tuning*.

É importante destacar que, na abordagem aqui apresentada, foi adotada apenas a primeira estratégia descrita por [Farias 2019], em que foi realizado o treinamento e gerado um modelo a partir do *dataset* SQuAD 1.1. Também foram realizados ajustes na taxa de aprendizagem e no número de épocas de treinamento do modelo, com o objetivo de melhorar seu desempenho.

Por fim, na etapa (iv), foram realizados os testes e validação do modelo, tanto com as questões (perguntas) originais do *corpus*, como também foram elaboradas novas perguntas, trocando termos por seus sinônimos, ou mesmo alterando a ordem dos termos, parafraseando a pergunta original. Os resultados desses testes foram estruturados em dois *datasets*, sendo um para as questões originais e o outro para as questões parafraseadas. Esses *datasets* foram definidos com as seguintes colunas: *question, title, paragraphs, answer, iteration e score*.

É importante observar que, nesta fase, o método *predict* teve seus parâmetros *retriever_score_weight = 0.51* e *n_predictions = 20* ajustados a partir de alguns experimentos. Com o primeiro ajuste (*retriever_score_weight*), o modelo apresentou um melhor desempenho, enquanto que o segundo (*n_predictions*) foi definido com o objetivo de identificar se o modelo chegava a utilizar todo o intervalo retornado como saída, para inferir a resposta correta, conforme detalhado na Seção 4. Com os ajustes, ao submeter uma pergunta, o modelo devolve um *array*⁴ contendo vinte tuplas com as respostas. Então, com o objetivo de salvar em um *dataset* as questões corretas devolvidas pelo modelo, para cada questão submetida como entrada ao modelo, foi realizada uma busca no *array* devolvido, procurando identificar a existência de uma tupla com a pergunta (*title*) correspondente.

3.2. Materiais e Método para o Segundo Cenário

Foi utilizada a mesma base de dados empregada no primeiro cenário, conforme detalhado na Seção 3.1. Para acesso ao modelo GPT, foi necessário um cadastro prévio na plataforma⁵ OpenAI e a criação de uma chave de acesso. Posteriormente, foi executada a API via código em Python para realização dos experimentos, com a seguinte configuração de parâmetros: *model = text-davinci-003; temperature = 0; max_tokens = 100; top_p = 1; frequency_penalty = 0; e, presence_penalty = 0*. Diferentemente do primeiro cenário, que retornava uma lista (*array*) com 20 respostas para cada pergunta, no segundo cenário, o modelo retorna por padrão uma única resposta para cada pergunta fornecida como entrada.

³Força de trabalho humana que executa tarefas virtualmente.

⁴Corresponde a uma forma de armazenar dados estruturados de um mesmo tipo.

⁵Disponível em: <https://platform.openai.com/>. Acesso em: 17 de março de 2023.

4. Resultados e Discussões

4.1. Para o Primeiro Cenário

Para ilustrar o desempenho do modelo no primeiro cenário foram selecionados dois experimentos: o primeiro utilizando as perguntas da própria base; e, o segundo utilizando as perguntas parafraseadas, conforme detalhado a seguir.

4.1.1. Utilizando Exatamente a Mesma Pergunta da Base

Considerando a pergunta P1: “*What is the role of a prototype program in problem solving?*”, em que o texto original da pergunta foi fornecido como entrada para o modelo treinado, o resultado obtido como resposta do modelo foi um *array* de tuplas, conforme descrito na etapa (iv) do método aqui apresentado, no qual os dados de interesse foram as perguntas (questões) (*title*) e as respostas (*paragraphs*), da seguinte forma:

- (P1.1, R1.1) = P1.1: “*What is the main advantage associated with function arguments that are passed by reference?*”, R1.1: “*It avoids making copies of large data structures when calling functions.*”
- (P1.2, R1.2) = P1.2: “*What is the role of a prototype program in problem solving?*”, R1.2: “*To simulate the behaviour of portions of the desired software product.*”

Observa-se que a pergunta consultada, P1, foi listada como uma das perguntas retornadas pelo modelo - o que pode ser constatado em (P1.2, R1.2) - em que para essa pergunta, a resposta correta foi apresentada na segunda iteração. E, observando que o modelo pode ser ajustado para retornar até 20 pares de perguntas e respostas, então considera-se tal consulta como uma classificação correta. Logo, após ajustar o modelo para fornecer como saída uma lista desses 20 pares de perguntas e respostas, ao passar como entrada para o modelo todas as 87 perguntas da base, o modelo classificou corretamente 54 questões, totalizando 62% de acurácia. Após a análise das saídas fornecidas pelo modelo, conforme ilustrado na Figura 1, foram identificados os seguintes resultados: o número de respostas corretas identificadas logo na primeira iteração foi 18; da segunda à quarta iteração, o modelo acertou 21; e da 5ª à 12ª, o modelo acertou 15. Cabe observar que a partir da 12ª iteração o modelo não retornou resposta correta.

4.1.2. Utilizando a Pergunta Parafraseada

Considerando a pergunta original, P2 = “*What is the role of a prototype program in problem solving?*”, utilizando uma ferramenta *Web*⁶ para parafrasear automaticamente as questões, encontrou-se a pergunta P2’ = “*What function does a prototype program serve in solving issues?*”, em que tal texto parafraseado P2’ foi fornecido como entrada para o modelo treinado, e o resultado obtido foi uma lista com os seguintes pares de perguntas e respostas:

- (P2.1, R2.1) =
P2.1: “*What is the difference between a constructor and a function?*”,
R2.1: “*It avoids making copies of large data structures when calling functions.*”

⁶Disponível em: <https://quillbot.com/>. Acesso em: 08 de Agosto. de 2022.

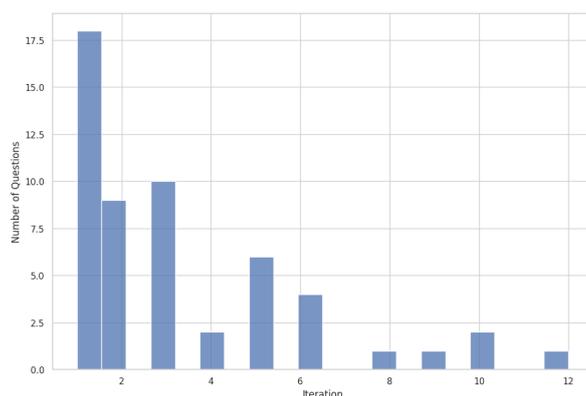


Figura 1. Dispersão das perguntas originais

- (P2.2, R2.2) = P2.2: *“What is the difference between an array declared as static, and one that is not?”*, R2.2: *“The arrays declared as static live throughout the life of the program; that is, they are initialized only once, when the function that declares the array it is first called.”*
- Os demais itens foram omitidos para simplificar a leitura.
- (P2.8, R2.8) =
P2.8: *“What is the role of a prototype program in problem solving?”*,
R2.8: *“To simulate the behaviour of portions of the desired software product.”*

Para avaliar se o modelo retornou o par de pergunta e resposta correspondente, foi utilizada a questão original P2 ao invés de P2' na verificação, uma vez que as questões parafraseadas não constam na base. Assim, verificada a similaridade entre a questão original e a questão retornada pelo modelo, considerou-se a listagem de P2.1 à P2.8 como uma classificação correta, uma vez que P2.8 é correspondente a P2.

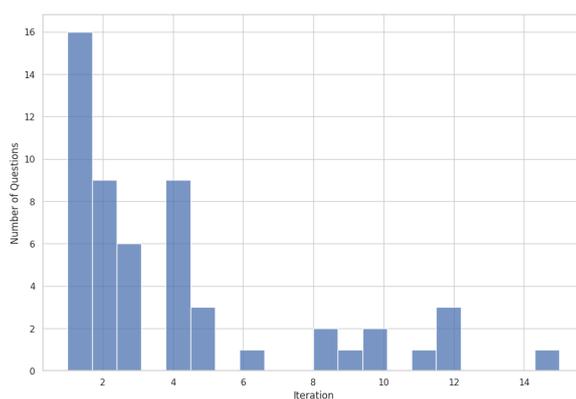


Figura 2. Dispersão das perguntas parafraseadas

No presente experimento, observou-se que o modelo apresentou um comportamento semelhante ao primeiro experimento, ou seja, ao se fornecer como entrada as 87 questões parafraseadas, o modelo classificou 54 questões corretas, e 33 incorretas, também alcançando 62% de acurácia. Após a análise das saídas fornecidas pelo modelo, foram identificados os seguintes resultados: conforme ilustrado no Figura 2, a partir das questões parafraseadas, o número de respostas corretas identificadas na primeira iteração

foi 16 (um pouco abaixo dos 18 obtidos no primeiro experimento); da 2ª à 4ª iteração, o modelo acertou 24 (um pouco acima dos 21 obtidos no primeiro experimento); e, da 5ª à 15ª, o modelo acertou 14 (um pouco abaixo dos 15 obtidos no primeiro experimento).

4.2. Para o Segundo Cenário

Utilizando a mesma base de dados do primeiro cenário, com o modelo GPT configurado conforme os parâmetros apresentados na seção anterior, obteve-se o seguinte desempenho, sobre as 87 perguntas:

- houve acerto em 74 das respostas, ou seja, aproximadamente 85% de acurácia. Além disso, é importante informar que as respostas apresentaram boa qualidade em termos de quantidade de informação e relevância; Por exemplo, dada a pergunta da base: *“What is the role of a prototype program in problem solving?”*, a resposta da base: *“To simulate the behaviour of portions of the desired software product.”*, a resposta do ChatGPT foi: *“A prototype program is a preliminary version of a software program that is used to test and demonstrate the concept of the program. It is used to help solve problems by providing a working model of the program that can be tested and modified. This allows developers to identify and fix any issues before the program is released.”*, o que foi classificado como uma resposta correta.
- 04 (quatro) respostas foram incorretas. Contudo, acredita-se que, caso fossem fornecidas mais informações contextuais na descrição da pergunta, o modelo provavelmente apresentaria respostas corretas. Por exemplo, para a pergunta: *“What is a tree?”*, a resposta do modelo foi *“A tree is a woody perennial plant, typically having a single stem or trunk growing to a considerable height and bearing lateral branches at some distance from the ground.”*, o que foi classificado como resposta incorreta; e,
- 09 (nove) respostas incompletas. Acredita-se que com um melhor ajuste nos parâmetros do modelo, tais respostas poderiam vir completas. Segue, um exemplo de resposta incompleta: *“The main advantages associated with object-oriented programming are:”*. Acredita-se que com uma melhor configuração dos parâmetros do modelo, a resposta poderia vir completa.

5. Considerações Finais

O presente trabalho apresentou e discutiu dois cenários de estudos de caso com utilização de *Question Answering (QA)*, sob uma base de dados com 87 questões relacionadas ao ensino de programação de computadores. No primeiro cenário, foram realizados ajustes no modelo inicialmente proposto por [Farias 2019] e, após experimentos submetendo tanto questões oriundas da base de treinamento quanto questões parafraseadas, o modelo ajustado alcançou em ambos cenários 62% de acurácia. No segundo cenário, utilizando o modelo GPT e a mesma base de dados com 87 questões, foi obtido 85% de acurácia.

Como indicação de trabalhos futuros, pretende-se disponibilizar a abordagem de QA do presente trabalho em um sistema Web, permitindo que o estudante possa acessar de forma distribuída e assíncrona tais conteúdos, e realizar consultas em linguagem natural por meio de um campo de texto, e assim, receber como resposta a listagem de conteúdos relacionados ao domínio de estudos em questão. Um outro aspecto interessante a ser investigado é verificar o desempenho de um ajuste do modelo de [Farias 2019], aplicado a

bases de dados com conteúdos de outras disciplinas. Ainda como perspectiva futura, tem-se alguns desdobramentos que podem ampliar a análise do presente trabalho, a exemplo de explorar as questões que o modelo não encontrou respostas. Uma possibilidade seria tentar ajustar o modelo com uma base maior, realizar outros ajustes por meio de *transferring learning* e *fine-tuning*, ou construir uma base com um título e um contexto (artigo, capítulo de livro) com conteúdos relacionados às respectivas questões.

Por fim, acredita-se que a presente abordagem possa auxiliar tanto professores quanto estudantes no desempenho de suas atividades e no processo de ensino-aprendizagem de programação de computadores, ao oferecer um ambiente de estudos focado nos objetivos de uma determinada disciplina, visando a garantia de que as consultas dos alunos retornem conteúdos diretamente associados ao que for lecionado pelo professor.

Referências

- Bispo Jr., E. L., Raabe, A., Matos, E. d. S., Maschio, E., Barbosa, E. F., Carvalho, L. G., Bittencourt, R. A., Duran, R. S., e Falcão, T. P. d. R. (2020). Tecnologias na Educação em Computação: Primeiros referenciais. *RBIE*, 28:509–527.
- Brown, T. et al. (2020). Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.
- Farias, A. M. (2019). How to create your own question-answering system easily with Python. Disponível em: <https://towardsdatascience.com/how-to-create-your-own-question-answering-system-easily-with-python-2ef8abc8eb5>. Acesso em: 27 de maio de 2023.
- Gil, A. C. et al. (2017). *Como elaborar projetos de pesquisa*, volume 6. Atlas São Paulo.
- Jurafsky, D. e Martin, J. H. (2023). *Speech and Language Processing*. Prentice Hall, 3 edition. Disponível em <https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf>.
- Lende, S. P. e Raghuvanshi, M. (2016). Question answering system on education acts using NLP techniques. In *2016 world conference on futuristic trends in research and innovation for social welfare (Startup Conclave)*, pages 1–6. IEEE.
- Mohler, M., Bunescu, R., e Mihalcea, R. (2011). Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 752–762.
- Rajpurkar, P., Zhang, J., Lopyrev, K., e Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.
- Roberts, A., Raffel, C., e Shazeer, N. (2020). How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*.
- Soares, M. A. C. e Parreiras, F. S. (2020). A literature review on question answering techniques, paradigms and systems. *Journal of King Saud University-Computer and Information Sciences*, 32(6):635–646.
- Swathilakshmi, V. et al. (2021). Student suite+, a closed domain question answering system for educational domain. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 12(10):3168–3172.