تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

تم استلام الورقة العلمية في	2026/04/15	Received
تم قبول الورقة العلمية في	2026/05/09	Accepted
تم نشر الورقة العلمية في	2026/05/10	Published

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

أ. أسماء عبد الله الكريك
كلية التقنية الصناعية – مصراته –ليبيا
Asma.alkraik2025@gmail.com

الملخص:

تتناول هذه الورقة البحثية تطوير نظام متكامل لمعالجة النصوص العربية والإنجليزية بالاعتماد على تقنيات معالجة اللغة الطبيعية (NLP) والنماذج اللغوية العميقة، بهدف تصحيح الأخطاء النحوية والإملائية، واستعادة علامات الترقيم، وإزالة التكرار غير الضروري، مع الحفاظ على المعنى الأصلي للنص. كما يتضمن النظام وحدة لتصنيف النصوص إلى فئتين هي: الطبية، القانونية باستخدام نموذج تصنيف متعدد اللغات.

هدفت الدراسة إلى تقييم النظام من جانبين رئيسيين: جودة التصحيح اللغوي، وأداء الاستدلال الحاسوبي في بيئتي المعالجة المركزية (CPU) والمعالجة الرسومية GPU باستخدام CUDA، مع مقارنة أسلوبي التنفيذ التسلسلي والمتوازي. وقد أُجريت التجارب على مجموعات نصية عربية وإنجليزية بأحجام مختلفة، وتم تقييم جودة المخرجات باستخدام مؤشري BLEU وGLEU.

أظهرت النتائج تفوق النظام المقترح على النظام المرجعي، خاصة في النصوص العربية، حيث بلغ متوسط (0.7454) BLEU و(0.7529) GLEU، بينما بلغت القيم للنصوص الإنجليزية (0.8143) و(0.8214) على التوالي. كما بينت النتائج أن المعالجة المتوازية أسهمت في تقليل زمن التنفيذ بصورة ملحوظة في بيئتي CPU وGPU، مع أفضل أداء

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

عند استخدام أحجام دفعات متوسطة تراوحت بين 64 و128. وأثبتت الدراسة كذلك ثبات النتائج بين التنفيذ التسلسلي والمتوازي بنسبة تطابق بلغت 100%.

تشير النتائج إلى أن النظام المقترح يوفر حلاً فعالًا يجمع بين جودة التصحيح وسرعة التنفيذ، مما يجعله مناسبًا لتطبيقات لتدقيق اللغوي، وتحليل النصوص، ومعالجة البيانات كبيرة الحجم.

الكلمات المفتاحية: معالجة اللغة الطبيعية، تصحيح النصوص، التصنيف النصي، المعالجة المتوازية، المعالج المركزي، معالج الرسوميات، BLEU، GLEU.

Evaluating Performance and Accuracy of Bilingual (Arabic–English) Grammar Correction Models on CPU vs GPU Platforms: Sequential vs Parallel Processing

Asma Abdullah Alkraik
College of Industrial Technology – Misurata – Libya
Asma.alkraik2025@gmail.com

ABSTRACT:
This research paper presents an integrated system for processing Arabic and English texts using Natural Language Processing (NLP) techniques and deep language models. The system is designed to correct grammatical and spelling errors, restore punctuation, remove unnecessary repetition, and preserve the original meaning of the text. It also includes a multilingual text classification module that categorizes texts into six major domains: medical, legal, academic, news, entertainment, and social.

The study aimed to evaluate the proposed system from two main perspectives: linguistic correction quality and inference performance under two computing environments, Central Processing Unit (CPU) and Graphics Processing Unit (GPU with CUDA), while comparing sequential and parallel execution modes. Experiments were conducted on Arabic and English text datasets of

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

different sizes, and output quality was measured using BLEU and GLEU metrics.
The results showed that the proposed system outperformed the baseline approach, especially for Arabic texts, achieving average BLEU and GLEU scores of 0.7454 and 0.7529, respectively. For English texts, the system achieved 0.8143 BLEU and 0.8214 GLEU. In addition, parallel processing significantly reduced execution time in both CPU and GPU environments, with the best performance obtained using medium batch sizes between 64 and 128. The study also confirmed full consistency between sequential and parallel outputs, with a 100% matching rate.
These findings indicate that the proposed system provides an effective solution that combines high correction quality with efficient execution speed, making it suitable for proofreading, text analytics, and large-scale text processing applications.
Keywords: Natural Language Processing, Text Correction, Text Classification, Parallel Processing, CPU, GPU, BLEU, GLEU.

1. المقدمة:

في ظل التطور السريع في مجال الحوسبة الرقمية وانتشار المحتوى النصي متعدد اللغات، ازدادت الحاجة إلى أدوات دقيقة وفعالة لمعالجة وتصحيح النصوص، خاصةً النصوص العربية والإنجليزية نظرًا لأهميتهما الواسعة في مجالات أكاديمية وإعلامية وقانونية وطبية وغيرها. تُعد عمليات التصحيح اللغوي والنحوي جزءًا أساسيًا من تحسين جودة المحتوى النصي وضمان وضوحه ودقته، مما يعزز من مصداقيته وقابليته للاستخدام في التطبيقات المختلفة [1].

تواجه عمليات تصحيح النصوص ثنائية اللغة تحديات عدة تتمثل في اختلاف البنى اللغوية، والتباين في القواعد النحوية والإملائية بين اللغتين، بالإضافة إلى ضرورة استعادة علامات الترقيم بدقة والحفاظ على المعنى الأصلي للنصوص. علاوة على ذلك، يُعد تصنيف النصوص ضمن فئات موضوعية محددة خطوة مهمة لتعزيز القدرة على التحليل والمعالجة النصية، مما يدعم تطبيقات متقدمة مثل التدقيق اللغوي وأرشفة المحتوى وتحليل البيانات [2].

http://www.doi.org/10.62341/istj-vol38-2-ka14

تعتمد هذه الدراسة على استخدام نماذج متقدمة في مجال معالجة اللغة الطبيعية (Natural Language Processing – NLP)، تشمل نماذج تصحيح الأخطاء اللغوية القائمة على التعلم العميق، ونماذج تصنيف النصوص متعددة اللغات، مع دمج تقنيات استعادة الترقيم وإزالة التكرار. كما تُجري الدراسة مقارنة عملية بين بيئتين للحوسبة: بيئة المعالجة المركزية (CPU) وبيئة معالج الرسوميات (GPU) باستخدام تقنية (CUDA)، مع تقييم شامل للأداء من حيث جودة التصحيح ودقة التصنيف وسرعة التنفيذ [3].

تُعد مهمة التصحيح اللغوي والنحوي من المهام الجوهرية في معالجة اللغات الطبيعية، حيث تهدف إلى رفع جودة النصوص وجعلها أكثر دقة ووضوحًا. ومع تزايد حجم البيانات وتنوعها، برزت الحاجة إلى تسريع عمليات المعالجة دون التأثير سلبًا على جودة النتائج، خاصة في البيئات متعددة اللغات. وفي ظل التطور المتسارع في تقنيات التعلم العميق ونماذج اللغة الكبيرة، أصبح من الضروري تقييم أداء هذه النماذج في بيئات الحوسبة المختلفة، مثل وحدات المعالجة المركزية (CPU) ووحدات معالجة الرسوميات (GPU)، بالإضافة إلى مقارنة أساليب التنفيذ التسلسلي والمتوازي [4].

تشير الأبحاث الحديثة إلى أن استخدام وحدات معالجة الرسوميات (GPU) يمكن أن يحقق تسريعًا ملحوظًا في مهام معالجة النصوص، خصوصًا عند دمجها مع تقنيات المعالجة المتوازية. في المقابل، تظل وحدات المعالجة المركزية (CPU) أكثر شيوعًا في البيئات التقليدية بسبب مرونتها، رغم محدودية سرعتها في المهام التي تتطلب توازيًا عاليًا [5].

في هذا البحث، نقدم نموذجًا عمليًا لتصحيح النصوص العربية والإنجليزية باستخدام نماذج متخصصة لكل لغة، مثل (CAMeL-ArabART) لتصحيح اللغة العربية [6]، و(Happy Transformer) و (LanguageTool) لتصحيح اللغة الإنجليزية [7][8]. كما يتم دمج نموذج التصنيف متعدد اللغات (mDeBERTa) لتحديد مجال النص (طبي، قانوني، أكاديمي، إخباري، ترفيهي، اجتماعي) [9]. وقد تم تطبيق الدراسة

http://www.doi.org/10.62341/istj-vol38-2-ka14

باستخدام خوارزميات التنفيذ التسلسلي والمتوازي على بيئتي المعالج المركزي (CPU) ومعالج الرسوميات (GPU) لتقييم الأداء من حيث زمن المعالجة وجودة النتائج.

تهدف الدراسة إلى تقديم نظام متكامل قادر على تصحيح وتصنيف النصوص الطويلة ثنائية اللغة بكفاءة عالية، مع تقديم توصيات عملية بالتحول نحو استخدام الحوسبة المتوازية على معالجات الرسوميات لتحقيق أقصى استفادة من قدرات الحوسبة الحديثة وتحسين الأداء في معالجة اللغات الطبيعية.

2. الدراسات السابقة:

تناولت العديد من الدراسات السابقة مسألة تسريع مهام معالجة اللغة الطبيعية باستخدام تقنيات الحوسبة المتوازية، إلا أنها اختلفت في المنهجيات وبيئات التنفيذ.

في عام 2012، أشار Ghorpade وآخرون [5] إلى أن استخدام وحدات معالجة الرسوميات (GPU) مع بنية CUDA يتيح تسريعًا ملحوظًا في العمليات الحسابية المتوازية مقارنة بالمعالجات التقليدية(CPU) ، وهو ما يجعل GPU خيارًا فعالًا للمهام كثيفة الحساب مثل البحث النصي والتوافق الدلالي في أنظمة NLP.

في عام 2012، اقترح **Ghorpade–Aher وآخرون** دراسة قارنت أداء تطبيقات معالجة النصوص بين وحدات معالجة الرسوميات (GPU) والمعالجات أحادية ومتعددة النواة، وأظهرت النتائج قدرة (GPU) على تسريع العمليات المعتمدة على البحث النصي والتوافق الدلالي مقارنة بالمعالجة التقليدية علىCPU [5].

في عام 2013، بيّنت دراسات حول أنظمة معالجة اللغة الطبيعية المبنية على بنى موزعة أهمية التوازي في تحسين الأداء، حيث طُوِّرت منصات تحليل نصوص متعددة الخوادم مثل PyPLN التي تسمح بتوزيع مهام معالجة البيانات عبر عدة وحدات تنفيذ، مما يسهم في تقليل زمن المعالجة مقارنة بالتنفيذ التسلسلي التقليدي عند التعامل مع مجموعات نصية ضخمة[10] .

وفي عام 2017، اقترح **Argueta و Chiang** تنفيذ خوارزميات لغوية إحصائية مثل **Viterbi**و **Forward–Backward** باستخدام GPU، وحققوا تسريعات وصلت إلى

http://www.doi.org/10.62341/istj-vol38-2-ka14

(5.2×) مقارنة بالتنفيذ علىCPU، مما يبرز قابلية الخوارزميات اللغوية للاستفادة من المعالجة المتوازية [11].

وفي عام 2018، قدّم الباحثان امتدادًا للدراسة السابقة عبر تطوير مركّب **Finite–State Transducer (FST)** يعمل على (GPU)، وحقق تسريعًا وصل إلى **(6×)** مقارنة بالتنفيذ التسلسلي، مما عزز دور GPU في الأنظمة اللغوية واسعة النطاق [12].

كما تناولت دراسات حديثة أداء وحدات المعالجة المركزية والرسومية في مهام **الاستدلال (Inference)**، حيث أظهرت أن GPU تحقق أفضلية واضحة عند التعامل مع نماذج عميقة وكبيرة مثل **(Transformers)**، في حين قد يكون CPU أكثر كفاءة في النماذج الصغيرة أو عند محدودية البيانات [13].

وفيما يخص التصحيح اللغوي، ركزت عدة أبحاث على تصحيح النصوص الإنجليزية باستخدام نماذج مثل **GECToR** و**T5** [7][14]، بينما ظهرت دراسات محدودة نسبيًا للغة العربية مثل أعمال **CAMeL–Lab** التي قدمت نماذج متقدمة لمعالجة وتصحيح النصوص العربية [6].

ومع ذلك، لا توجد (حسب علمنا) حلول متكاملة تجمع بين (هذا ما تسعى هذه الدراسة إلى تحقيقه):

- التصحيح النحوي والدلالي للغتين العربية والإنجليزية.
- التصنيف الموضوعي للنص بعد التصحيح.
- مقارنة أداء التنفيذ التسلسلي والمتوازي على بيئتي (CPU) و(GPU).

3. تقنيات معالجة اللغة الطبيعية (NLP) والنماذج اللغوية العميقة المستخدمة:

1.3 نموذج CAMeL–Lab/arabart–qalb15–gec–ged–13 للتصحيح العربي:

يُعد نموذج CAMeL–Lab/arabart–qalb15–gec–ged–13 نموذجًا متقدمًا لتصحيح الأخطاء اللغوية في اللغة العربية الفصحى الحديثة (MSA)، حيث تم تطويره عبر تخصيص (Fine–tuning) نموذج AraBART القائم على بنية Sequence–to–Sequence المعتمدة على Transformer [15].

http://www.doi.org/10.62341/istj-vol38-2-ka14

تم تدريب النموذج باستخدام بيانات (QALB–2015) بعد إخضاعها للمعالجة الشكلية (Morphological Preprocessing)، وهي مجموعة بيانات متخصصة في تصحيح الأخطاء النحوية والإملائية والصرفية الشائعة في النصوص العربية [16].

يمتاز هذا النموذج بتكامله مع نموذج كشف الأخطاء النحوية (GED) متعدد الفئات (13 فئة)، وتعتمد بعض نماذج تصحيح الأخطاء اللغوية الحديثة على تزويد النموذج بمعلومات صريحة حول نوع الخطأ اللغوي، كما هو الحال في النماذج القائمة على التوسيم مثل **GECToR**، حيث أظهرت الدراسات أن هذا النهج يؤدي إلى تحسين ملحوظ في دقة التصحيح مقارنة بالنماذج التقليدية، مما يجعله مناسبًا لأنظمة التدقيق اللغوي وأدوات المساعدة الكتابية .[15].

من ناحية أخرى، يعتمد نموذج **AraBART** في بنيته الأساسية على معمارية **BART**، وهي بنية قائمة على **Transformer**من نوع **Encoder–Decoder**، وتُستخدم بكفاءة في مهام توليد النصوص مثل الترجمة، التلخيص، وتصحيح الأخطاء [17]. ويُعد AraBART تكييفًا لهذه البنية مع اللغة العربية من خلال تدريبه المسبق على بيانات عربية واسعة تراعي الخصائص اللغوية والنحوية للغة، مما يجعله مناسبًا لمهام المعالجة اللغوية العربية المتقدمة [15].

إضافةً إلى ذلك، يعتمد النموذج على المعالجة الشكلية قبل التدريب، مما يساهم في تبسيط التمثيل الصرفي للكلمات العربية وتحسين قدرة النموذج على فهم البنية اللغوية للنص، وهو ما ينعكس إيجابًا على دقة التصحيح [16].

يتميز النموذج بما يلي:

- القدرة على تصحيح طيف واسع من الأخطاء اللغوية في العربية الفصحى.
- الدمج بين التعلّم العميق والمعالجة الشكلية.
- ملاءمته لتطبيقات التدقيق اللغوي، والتعليم، وأنظمة المساعدة الكتابية.

مثال:

- الجملة الأصلية:

"دهبت الى المدرسة امس ولم ادرس جيدا"

- الجملة بعد التصحيح (متوقع):

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

"ذهبت إلى المدرسة أمس ولم أدرس جيدًا"

2.3 نموذج vennify/t5-base-grammar-correction للتصحيح انجليزي:

يعتمد نموذج (vennify/T5-base-grammar-correction) على معمارية (T5) (Text-to-Text Transfer Transformer)، وهي معمارية موحدة تقوم بتحويل جميع مهام معالجة اللغة الطبيعية إلى مهمة توليد نص من نص [7]. تم تطوير معمارية (T5) من قبل Google، وتعتمد على (Transformer Encoder–Decoder)، حيث يستقبل النموذج جملة نصية (قد تحتوي على أخطاء) ويولّد جملة مصححة كنص مخرج [1].

تم تخصيص النموذج (Fine-tuning) باستخدام مجموعات بيانات متخصصة في تصحيح الأخطاء النحوية في اللغة الإنجليزية، من أبرزها:

- **FCE Dataset** الخاصة بمتعلمين اللغة الإنجليزية [18].
- مجموعات بيانات **CoNLL-2013** و **CoNLL-2014** لتصحيح الأخطاء النحوية [19].

يسمح هذا التخصيص للنموذج بالتعامل مع أخطاء متعددة تشمل الأخطاء الإملائية والنحوية وعلامات الترقيم، مع الحفاظ على المعنى الأصلي للنص. ويتميز النموذج بـ:

- قدرته على تصحيح أخطاء متشابكة ومعقدة.
- جودة عالية في النص الناتج.
- استخدام تقنيات توليد متقدمة مثل Beam Search [7].
- ملاءمته لأدوات التدقيق اللغوي وبرامج تحرير النصوص والمساعدات التعليمية.

مثال:

- الجملة الأصلية:

"She no went to the market yesterday."

- الجملة بعد التصحيح:

"She did not go to the market yesterday."

3.3 نموذج MoritzLaurer/mDeBERTa-v3-base-mnli-xnli لتصنيف النص المتعدد اللغة:

http://www.doi.org/10.62341/istj-vol38-2-ka14

يُعد نموذج (mDeBERTa-v3) نموذجًا متعدد اللغات مبنيًا على بنية (DeBERTa) المطورة من مايكروسوفت، ويُستخدم لفهم العلاقات الدلالية بين الجمل ضمن إطار Natural Language Inference (NLI) [9].

ويمتاز النموذج بقدرته على تنفيذ مهام (Zero-Shot Classification) دون الحاجة إلى تدريب مسبق على الفئات المستهدفة [20].

تم تدريب النموذج على مجموعتي بيانات رئيسيتين:

- MNLI، وهي مجموعة بيانات إنجليزية متعددة المجالات [21].
- XNLI، وهي امتداد متعدد اللغات يدعم أكثر من 15 لغة، من بينها العربية والإنجليزية [22].

تعتمد آلية التصنيف الصفري على إعادة صياغة مهمة التصنيف كمهمة استدلال لغوي، حيث يتم تمثيل كل فئة على شكل فرضية(Hypothesis) ، ويحدد النموذج درجة الدعم (Entailment) بين النص والفئة، ويتم اختيار الفئة ذات أعلى احتمال [20].

تتميز هذه المنهجية بما يلي:

- عدم الحاجة إلى بيانات تدريب إضافية لكل فئة.
- مرونة عالية في التعامل مع مجالات متعددة.
- كفاءة خاصة في البيئات ثنائية اللغة بفضل التدريب على (XNLI) [22].

4. بيئة المعالجة المعتمدة:

تعتمد الدراسة على المقارنة بين المعالجة باستخدام **المعالج المركزي (CPU)** والمعالجة باستخدام **معالج الرسوميات (GPU)**. يتميز CPU بعدد محدود من الأنوية ذات سرعة عالية، وهو مناسب للمهام المتسلسلة والمرونة البرمجية، لكنه أقل كفاءة في المهام كثيفة الحساب مثل تشغيل نماذج Transformer الكبيرة [10].

في المقابل، يحتوي (GPU) على آلاف الأنوية المصممة للمعالجة المتوازية، ويُستخدم مع تقنية (CUDA) لتسريع الحسابات في أطر التعلم العميق مثل PyTorch

http://www.doi.org/10.62341/istj-vol38-2-ka14

وTensorFlow. وقد أظهرت تقارير NVIDIA أن GPU يمكن أن يكون أسرع بعشرات المرات مقارنة بـ CPU في بعض مهام التعلم الآلي [23].

5.أنماط المعالجة المعتمدة:

تعتمد المعالجة التسلسلية على تنفيذ المهام واحدًا تلو الآخر، وهي سهلة التنفيذ لكنها تعاني من زمن كلي مرتفع عند التعامل مع بيانات كبيرة [10].

أما المعالجة المتوازية، فتعتمد على توزيع المهام على عدة Threads أو أنوية، مما يؤدي إلى تقليل زمن التنفيذ الكلي وتحسين الأداء، خاصة في حالات القراءة والكتابة ومعالجة الدُفعات النصية الكبيرة [5].

6. منهجية البحث (Research Methodology):

يعتمد هذا البحث على تطوير إطار برمجي موحّد ومتكامل لتصحيح وتصنيف النصوص ثنائية اللغة (العربية والإنجليزية)، باستخدام تقنيات معالجة اللغة الطبيعية (NLP) ونماذج التعلم العميق. يتضمن الإطار مراحل متعددة تبدأ من التحميل والتحضير، ثم المعالجة اللغوية والتصحيح والتصنيف، مع تقييم أدائه على بيئتي المعالجة **بالمعالج المركزي (CPU)** والمعالجة **بمعالج الرسوميات (CUDA)**، وذلك في حالتي **المعالجة التسلسلية** و**المعالجة المتوازية** .يهدف هذا التقييم إلى قياس تأثير بنية المعالجة والبيئة الحاسوبية على زمن التنفيذ ودقة النتائج، وتوضح هذه المنهجية الخطوات التنفيذية المعتمدة في الدراسة، كما يلي:

1.6 إعداد بيئة العمل:

تم إعداد بيئة البرمجة باستخدام لغة (Python) مع الاعتماد على أحدث إصدارات المكتبات الداعمة لمعالجة اللغات الطبيعية والتعلم العميق، وبشكل خاص مكتبة (PyTorch) التي تتيح تنفيذ النماذج على كل من (CPU) و(GPU).

تم تنفيذ التجارب في وضعين رئيسيين:

- وضع (CPU): تنفيذ العمليات على المعالج المركزي باستخدام خيط واحد (تسلسلي) أو عدة خيوط (متوازي).

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

- وضع CUDA (GPU): تنفيذ عمليات الاستدلال باستخدام معالج الرسوميات للاستفادة من التوازي العميق الذي توفره بنية (CUDA).

2.6 إعداد البيانات وتحميل النصوص:

اعتمدت هذه الدراسة على مجموعة بيانات موسعة تم إعدادها من مصادر عامة مفتوحة ومتاحة للاستخدام البحثي، بهدف ضمان تنوع المحتوى النصي وتمثيل بيئات لغوية واقعية لكل من اللغتين العربية والإنجليزية. وقد شملت مصادر البيانات ما يلي:

- **نصوص قانونية منشورة رسميًا** مثل اللوائح، القوانين، القرارات التنظيمية، والأحكام المتاحة عبر المواقع الحكومية والمصادر العامة .
- **مقالات طبية وتعليمية مفتوحة** تشمل مواد توعوية، مقالات صحية، ونصوص تعليمية منشورة عبر منصات معرفية مفتوحة .
- نصوص عامة متنوعة لدعم اختبار التصنيف الموضوعي في مجالات متعددة .

بعد جمع البيانات، تم تنظيف النصوص ومعالجتها أوليًا، ثم تحويلها إلى ملفات نصية بصيغة **TXT** ليتم استخدامها داخل بيئة الاختبار البرمجية.

ولأغراض التقييم الموضوعي والدقيق لنماذج التصحيح اللغوي، تم إنشاء **نسختين من كل نص** كما يلي:

1. **النسخة المرجعية الصحيحة:**

   وهي النص الأصلي بعد مراجعته لغويًا واعتماده كنص صحيح مرجعي.

2. **نسخة تحتوي على أخطاء مصطنعة تحاكي الأخطاء الواقعية**، وتم توليدها آليًا بإدخال أنماط متنوعة من الأخطاء الشائعة، من بينها :

- أخطاء إملائية .
- حذف أو تغيير علامات الترقيم .
- تكرار كلمات أو أحرف .
- أخطاء نحوية بسيطة .
- استبدال الحروف المتشابهة شكليًا أو صوتيًا .
- حذف المسافات أو إضافتها في مواضع خاطئة .

http://www.doi.org/10.62341/istj-vol38-2-ka14

وقد استُخدمت النسخة المرجعية الصحيحة كأساس للمقارنة عند حساب مؤشري **BLEU** و **GLEU**، لقياس مدى قدرة النماذج على استعادة النص الصحيح من النسخة المشوشة. كما تم توزيع البيانات بين نصوص عربية وإنجليزية وبأطوال مختلفة، لضمان عدالة التقييم في بيئات المعالجة المختلفة (CPU/GPU) وفي نمطي التنفيذ التسلسلي والمتوازي.

ويتم كشف لغة كل ملف نصي تلقائيًا باستخدام مكتبة (langdetect)، مما يتيح:

- دعم النصوص ثنائية اللغة دون تدخل يدوي.
- توجيه النص إلى نموذج التصحيح المناسب حسب اللغة.

3.6 آليات التنفيذ حسب بيئة المعالجة:

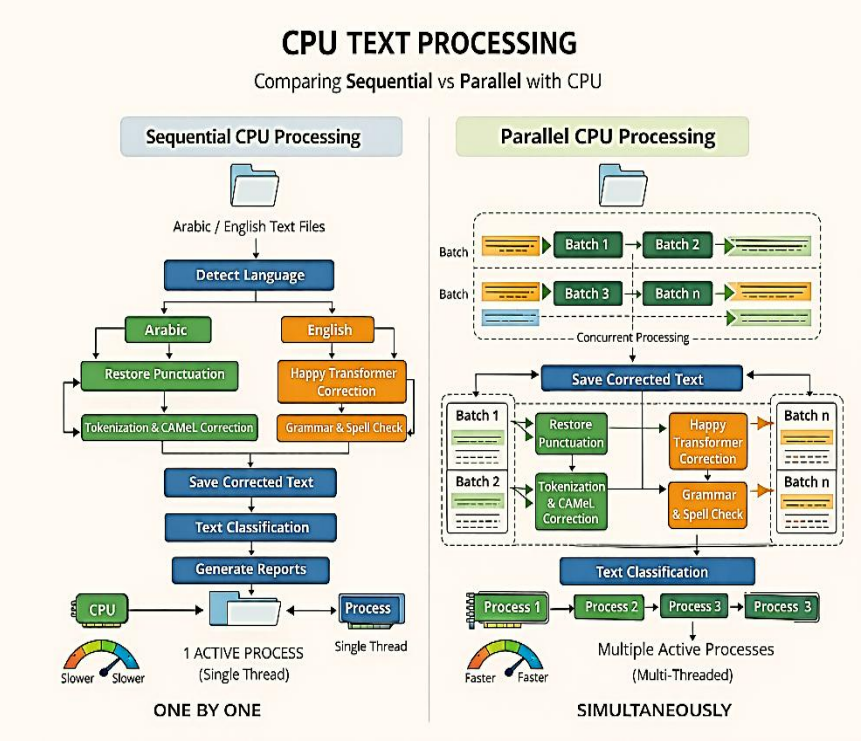
لفهم الفروقات في الأداء بين أنماط التنفيذ المختلفة، تم تصميم مخططات توضيحية تبين مسارات تدفق البيانات وآليات جدولة العمليات في بيئتي المعالجة CPU وGPU، وذلك في حالتي التنفيذ التسلسلي والمتوازي.

يوضح الشكل (1) آليات التنفيذ في بيئة المعالجة CPU لكل من النمط التسلسلي والمتوازي. في الحالة التسلسلية، يتم تنفيذ مراحل المعالجة (كشف اللغة، الاسترجاع النحوي، التصحيح، التصنيف، وتوليد التقارير) ضمن خيط تنفيذ واحد (Single Thread)، حيث تتم معالجة كل ملف على حدة قبل الانتقال إلى الملف التالي. يؤدي هذا الأسلوب إلى استهلاك منخفض للموارد لكنه يحد من الاستفادة من تعدد الأنوية، مما ينعكس على زيادة زمن التنفيذ الكلي.

في المقابل، يبين الجزء الأيمن من الشكل اعتماد المعالجة المتوازية على عدة عمليات نشطة(Multi-Threaded / Multi-Process Execution) ، حيث يتم توزيع الملفات على أكثر من مسار تنفيذ في الوقت ذاته. يتيح ذلك تقليل زمن الانتظار وتحسين معدل الإنتاجية(Throughput) ، خاصة عند التعامل مع عدد كبير من الملفات. إلا أن هذا النمط قد يؤدي إلى زيادة في استهلاك الذاكرة وارتفاع كلفة التزامن (Synchronization Overhead)، مما يتطلب موازنة دقيقة بين عدد العمليات والأداء الفعلي.

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14



شكل 1. آليات التنفيذ في بيئة المعالجة CPU في حالتي المعالجة التسلسلية والمعالجة المتوازية

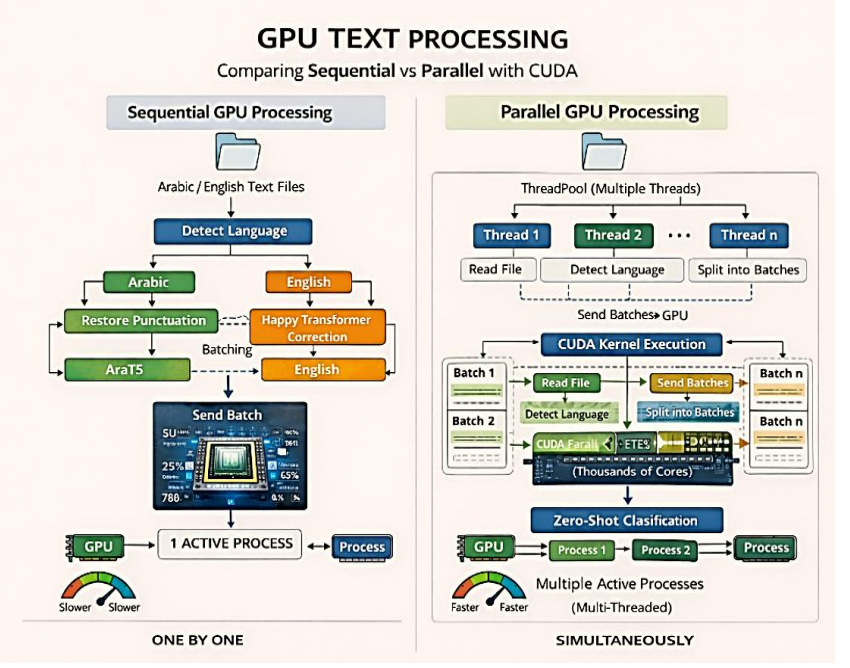
يبين الشكل (2) مسارات التنفيذ في بيئة GPU باستخدامCUDA ، مع مقارنة بين التنفيذ التسلسلي والتنفيذ المتوازي. في الوضع التسلسلي، يتم إرسال كل دفعة (Batch) إلى المعالج الرسومي بشكل منفصل، حيث تُنفذ العمليات داخل نواة CUDA واحدة في كل مرة. ورغم الاستفادة من التسريع العتادي، إلا أن عدم استغلال التوازي الكامل يؤدي إلى انخفاض في نسبة إشغال وحدة المعالجة الرسومية.(GPU Utilization)

أما في النمط المتوازي، فيتم تقسيم البيانات إلى دفعات متعددة تُرسل إلى GPU عبر آلية جدولة تعتمد على تعدد الخيوط في مستوىCPU ، مع تنفيذها داخلياً عبر آلاف الأنوية المتوازية في CUDA. يسمح ذلك بتحقيق درجة أعلى من استغلال الموارد الحاسوبية، وزيادة ملحوظة في معدل الإنتاجية، خصوصاً عند استخدام أحجام دفعات مناسبة. ومع

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

ذلك، فإن زيادة عدد الدفعات أو حجمها قد يؤدي إلى الوصول إلى نقطة تشبع (Saturation Point) نتيجة امتلاء الذاكرة الرسومية أو زيادة زمن الجدولة.



شكل 2. آليات التنفيذ في بيئة المعالجة GPU في حالتي المعالجة التسلسلية والمعالجة المتوازية

أولًا: المعالجة باستخدام (CPU – تنفيذ تسلسلي):

الفكرة الأساسية:

تنفيذ جميع خطوات المعالجة على نواة واحدة دون أي تداخل زمني.

الخطوات التنفيذية:

1. قراءة ملف نصي واحد من القرص.
2. كشف لغة النص (عربي / إنجليزي).
3. تقسيم النص إلى أسطر.
4. تمرير كل سطر إلى نموذج التصحيح المناسب:

http://www.doi.org/10.62341/istj-vol38-2-ka14

- العربية: نموذج تسلسلي (Seq2Seq).
- الإنجليزية: نماذج قائمة على T5 وأدوات لغوية.

5. انتظار انتهاء تصحيح السطر قبل الانتقال إلى السطر التالي.
6. إعادة تجميع النص المصحح.
7. تنفيذ تصنيف النص (طبي / قانوني).
8. حفظ النتائج.
9. الانتقال إلى الملف التالي.

ثانيًا: المعالجة باستخدام (CPU – تنفيذ متوازي):

الفكرة الأساسية:

تشغيل عدة ملفات في نفس الوقت باستخدام خيوط متعدد(Threads):

الخطوات التنفيذية:

1. جمع جميع الملفات النصية.
2. إنشاء ThreadPoolExecutor.
3. تخصيص ملف نصي مستقل لكل خيط.
4. كل خيط ينفذ داخليًا نفس خطوات المعالجة التسلسلية:
   - كشف اللغة.
   - تصحيح النص.
   - تصنيف النص.
   - حساب الإحصاءات.
5. تنفيذ الخيوط بالتوازي.
6. تجميع النتائج النهائية وحفظها.

ثالثًا: المعالجة باستخدام ((GPU (CUDA – تنفيذ تسلسلي):

الفكرة الأساسية:

استخدام GPU لتسريع عمليات النموذج، مع معالجة الملفات واحدًا تلو الآخر.

http://www.doi.org/10.62341/istj-vol38-2-ka14

الخطوات التنفيذية:

1. تحميل النموذج ونقله إلى ذاكرة GPU.
2. قراءة ملف نصي واحد.
3. تقسيم النص إلى دفعات (Batching).
4. نقل الدفعات إلى ذاكرة GPU.
5. تنفيذ التصحيح داخل GPU.
6. إعادة تجميع النص المصحح.
7. تنفيذ تصنيف النص.
8. حفظ النتائج.
9. الانتقال إلى الملف التالي.

رابعًا: المعالجة باستخدام ((GPU (CUDA – تنفيذ متوازي):

الفكرة الأساسية:

الجمع بين التوازي على مستوى الملفات (Threads) والتوازي العميق داخل GPU.

الخطوات التنفيذية:

1. تحميل نماذج التصحيح والتصنيف مرة واحدة إلى ذاكرة GPU.
2. إنشاء مجموعة من الخيوط لمعالجة الملفات بالتوازي.
3. تخصيص ملف مستقل لكل خيط.
4. تقسيم النص داخل كل ملف إلى دفعات مناسبة.
5. إرسال الدُفعات إلى GPU لإجراء الاستدلال.
6. تنفيذ التصحيح والتصنيف بالتوازي داخليًا عبر CUDA.
7. إعادة النتائج إلى كل خيط.
8. تجميع النتائج النهائية.

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

4.6 مراحل المعالجة الداخلية:

1.4.6 كشف اللغة:

يتم تحديد لغة النص تلقائيًا باستخدام مكتبة (langdetect) لتوجيه النص إلى النموذج المناسب.

2.4.6 التصحيح اللغوي والنحوي

أ. **النصوص الإنجليزية:**

تمر بثلاث مراحل:

1. تصحيح نحوي أولي باستخدام نموذج ذكاء اصطناعي (GECToR أو T5).
2. تدقيق إضافي باستخدام (LanguageTool).
3. تصحيح إملائي نهائي مع الحفاظ على علامات الترقيم.

ب. **النصوص العربية:**

تعتمد على نموذج (AraT5) وتشمل:

1. استرجاع علامات الترقيم.
2. ترميز النص وتحديد الطول الأقصى.
3. تنفيذ التصحيح على (CPU) أو (GPU).
4. فك الترميز.
5. فلترة التغييرات الدلالية الكبيرة.
6. تنقية علامات الترقيم.
7. إرجاع النص المصحح النهائي.

3.4.6 تصنيف نوع النص:

تم استخدام نموذج (Zero-Shot Classification) المعتمد على (MoritzLaurer /mDeBERTa-v3-base-mnli-xnli) لتصنيف النصوص إلى:

- طبي
- قانوني

المخرجات:

http://www.doi.org/10.62341/istj-vol38-2-ka14

- نوع النص الأكثر احتمالًا.
- درجة الثقة بين (0–1).

7. التجارب والنتائج:

تم تنفيذ أربع تجارب أساسية على بيئتي المعالجة بالمعالج المركزي (CPU) والمعالجة بمعالج الرسوميات(CUDA)، وذلك في حالتي المعالجة التسلسلية والمعالجة المتوازية لكلا البيئتين لأعداد مختلفة للنصوص العربية وانجليزية مصنفة كنصوص قانونية وطبية، وقد تم اعتماد منهجين مستقلين للتقييم:

1. تقييم جودة النموذج اللغوية باستخدام BLEU و GLEU ودقة التصنيف.
2. تقييم أداء الاستدلال باستخدام الزمن الكلي، Throughput، واستهلاك الموارد.

حيث اعتمدت التجارب على جهاز حاسوب شخصي يمتلك المواصفات العتادية والبرمجية التالية:

- نظام التشغيل :Windows 11 (64–bit).
- وحدة المعالجة المركزية :(CPU) Intel 13th Gen Core i7 بسرعة 2.40 GHz.
- وحدة معالجة الرسوميات :(GPU) NVIDIA GeForce RTX 4070.
- المكتبات والبرمجيات المستخدمة :PyTorch، Transformers، pandas، python–docx، matplotlib، بالإضافة إلى مكتبات مساعدة أخرى.

تم استخدام هذه المنصة لتدريب النماذج ومعالجة البيانات وتنفيذ جميع عمليات التقييم، بما في ذلك حساب الدقة ومقاييس الأداء الأخرى. وقد حافظت التجارب على إعدادات موحدة لضمان الاتساق بين مختلف مراحل التنفيذ.

أ. تقييم جودة النموذج اللغوية (Model Capability Evaluation)

في هذا القسم يتم تقييم دقة التصنيف (درجة الثقة) وجودة النموذج اللغوية باستخدام مؤشري BLEU وGLEU، وذلك من خلال عرض عينات ممثلة من النصوص قبل التصحيح وبعده.

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

عينة لنص طبي عربي تم تصنيفه كنص **طبي** وقد كانت درجة الثقة 0.95 وكانت متقاربة في كلا البيئتين، وكانت جودة التصحيح BLEU=0.7884 و GLEU=0.7934:

النص قبل التصحيح:

تختلف استجابه الافراد للبكتيرية المنقولة عبر الطعام بشكل كبير من شخص لأخر فقد تظهر الاعراض فور تناول الطعام الملوت أو بعد فترة تمتد من ايام إلى أسابيع. وحتى في حالة تناول شخصين نفس الوجبة الملوثة، قد يصاب أحدهما بالمرض بينما يبقى الآخر سليمًا.

تزداد شده الأعراظ عند بعض الفئات الأكثر عرضة، مثل الأضفال الرضع وكبار السن والأشخاص الدين يعانون من ظعف في جهاز المناعة.

وقد تئودي الأعراض الشديدة للأمراض المنقولة عبر الطعام إلى مضاعفات خطيرة متل الجفاف، والتي قد تستدعي التدخل الطبى وضرورة دخول المستشفي.

النص بعد التصحيح:

تختلف استجابة الأفراد للبكتيريا المنقولة عبر الطعام بشكل كبير من شخص لآخر. فقد تظهر الأعراض فور تناول الطعام الملوث، أو بعد فترة تمتد من أيام إلى أسابيع. وحتى في حالة تناول شخصين نفس الوجبة الملوثة، قد يصاب أحدهما بالمرض، بينما يبقى الآخر سليما.

تزداد شدة الأعراض عند بعض الفئات الأكثر عرضة، مثل: الأطفال، الرضع، وكبار السن، والأشخاص الذين يعانون من ضعف في جهاز المناعة.

وقد تؤدي الأعراض الشديدة للأمراض المنقولة عبر الطعام إلى مضاعفات خطيرة، مثل: الجفاف، والتي قد تستدعي التدخل الطبي وضرورة دخول المستشفى.

عينة لنص قانوني عربي تم تصنيفه كنص **قانوني** وقد كانت درجة الثقة 0.95 وكانت متقاربة في كلا البيئتين وكانت جودة التصحيحBLEU=0.7607 و GLEU=0.7812:

النص قبل التصحيح:

بعد الاطلاع على

الاعلان الدستوري المئوقت في 3 اغسطس 2011 م وتعديلاتة.

القانون رقم 10 لسنه 2014 م في انتخاب مجلس النواب في المرحلة الانتقالية وتعديلاته.

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

القانون رقم 4 لسنة 2014 م في شأن النظام الداخلي لمجلس النواب.

القانون رقم 12 لسنة 2010 م بشأن اصدار قانون العمل ولائحته التنفيدية.

القانون رقم 43 لسنة 1974 م بشأن تقاعد العسكريين.

القانون رقم 13 لسنة 1980 م بشأن الضمان الاجتماعى وتعديله.

القانون رقم 5 لسنة 2013 م بإضافة حكم للقانون رقم 13 لسنة 1980 بشأن الضمان الاجتماعي

وعلى ما خلص اليه مجلس النواب في اجتماعه الطارئ المنعقد يوم الأربعاء، 1 جمادي الأولى، 1445 هـ الموافق 13 ديسمبر 2023م.

صدر القانون الآتي

المادة 1

تسري أحكام القانون رقم 5 لسنة 2013م بإضافة حكم للقانون رقم 13 لسنة 1980م بشأن الضمان الاجتماعي على المتقاعدين أيا كان تاريخ تقاعدهم وتسوى معاشاتهم وفقاً لأحكامه.

المادة 2

يعمل بهذا القانون من تاريخ صدوره، ويلغى كل حكم يخالف أحكامه، وتنشر بالجريده الرسمية.

النص بعد التصحيح:

بعد الاطلاع على هذا الموضوع.

الإعلان الدستوري المؤقت في 3 أغسطس 2011 م وتعديلاته.

القانون رقم 10 لسنة 2014 م في انتخاب مجلس النواب في المرحلة الانتقالية وتعديلاته.

القانون رقم 4 لسنة 2014 م في شأن النظام الداخلي لمجلس النواب.

القانون رقم 12 لسنة 2010 م بشأن إصدار قانون العمل ولائحته التنفيذية.

القانون رقم 43 لسنة 1974 م بشأن تقاعد العسكريين.

القانون رقم 13 لسنة 1980 م بشأن الضمان الاجتماعي : وتعديله.

القانون رقم 5 لسنة 2013 م بإضافة حكم للقانون رقم 13 لسنة 1980 بشأن الضمان الاجتماعي.

وعلى ما خلص إليه مجلس النواب في اجتماعه الطارئ المنعقد يوم الأربعاء 1 جمادي الأولى، 1445 هـ الموافق 13 ديسمبر 2023 م.

صدر القانون الآتي :

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

المادة 1.

تسري أحكام القانون رقم 5 لسنة 2013م بإضافة حكم للقانون رقم 13 لسنة 1980م بشأن الضمان الاجتماعي على المتقاعدين أيا كان تاريخ تقاعدهم، وتسوى معاشاتهم وفقا لأحكامه.

المادة 2.

يعمل بهذا القانون من تاريخ صدوره، ويلغى كل حكم يخالف أحكامه، وينشر بالجريدة الرسمية.".

عينة لنص طبي انجليزي تم تصنيفه كنص **طبي** وقد كانت درجة الثقة 0.74 وكانت متقاربة في كلا البيئتين وكانت جودة التصحيحBLEU=0.8264 و GLEU=0.8553:

النص قبل التصحيح:

Vaccinations is a fundamentals component of preventive healthcare playing a crucial role in protection children from serious diseases such measles and polio. In addition, there are vaccines for less severe or seasonal illnesses such as the annual influenza shot. Adhering to the recommended immunization schedule is essential to ensured that children receive the necessary doses at the appropriate time. This significantly reduces the risk of infection and contributes to maintaining their overall health and well-being.

النص بعد التصحيح:

Vaccinations are a fundamental component of preventive healthcare, playing a crucial role in protecting children from serious diseases such as measles and polio. In addition, there are vaccines for less severe or seasonal illnesses, such as the annual influenza shot. Adhering to the recommended immunization schedule is essential to ensure that children receive the necessary doses at the appropriate times. This significantly reduces the risk of infection and contributes to maintaining their overall health and well-being.

ويوضح الجدول (1) متوسط جودة التصحيح للنصوص العربية والإنجليزية في النظام المقترح مقارنة بالنظام المرجعي (Baseline) باستخدام مؤشري BLEU وGLEU، وذلك لتقييم فعالية النموذج بشكل موضوعي، وقد تم تم اعتماد نظام مرجعي (Baseline) يعتمد على أداة LanguageTool لتصحيح النصوص العربية والإنجليزية دون استخدام

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

نماذج التعلم العميق أو آليات المعالجة المتوازية. وتمت مقارنة النظام المرجعي مع النظام المقترح من حيث جودة التصحيح (BLEU وGLEU) لقياس مدى تقارب النصوص المصححة مع النصوص المرجعية.

جدول 1. مقارنة متوسط جودة تصحيح النصوص العربية والإنجليزية بين النظام المقترح والنظام المرجعي باستخدام مؤشري BLEU وGLEU

أداة القياس	النصوص العربية	النصوص الانجليزية
BLEU	0.7454	0.8143
GLEU	0.7529	0.8214
BLEU_Baseline	0.6871	0.8128
GLEU_Baseline	0.6980	0.8199

تُظهر النتائج أن النظام المقترح تفوق على النظام المرجعي، خصوصًا في معالجة النصوص العربية، حيث حقق تحسنًا واضحًا في قيم BLEU وGLEU. أما في النصوص الإنجليزية فقد كان التحسن طفيفًا، إلا أن النظام حافظ على أداء مرتفع مقارنة بالأداة التقليدية، مما يعكس استقراره وجودته عبر اللغتين.

ب. **تقييم أداء الاستدلال (Inference Performance Evaluation)**

في هذا القسم يتم عرض ومناقشة النتائج التجريبية لأداء النظام المقترح في معالجة النصوص ثنائية اللغة (العربية والإنجليزية).

ويهدف ذلك إلى تقييم فعالية أسلوب التنفيذ المتوازي مقارنة بالتنفيذ التسلسلي في بيئتي المعالجة المركزية (CPU) والرسومية (GPU **باستخدام CUDA**).

تم إجراء التجارب على ثلاثة أحجام مختلفة من البيانات (100، 500، 1000 نص) باستخدام حجم دفعة ثابت Batch Size = 128 في بيئةGPU، وذلك لضمان عدالة المقارنة وتحليل تأثير حجم البيانات على الأداء.

تركز هذه الدراسة على قياس زمن التنفيذ في النمطين التسلسلي والمتوازي، بالإضافة إلى تحليل نسبة التحسن واستهلاك الذاكرة في بيئة GPU.

http://www.doi.org/10.62341/istj-vol38-2-ka14

ويساعد هذا التقييم في فهم مدى كفاءة النموذج المقترح من حيث السرعة وقابلية التوسع (Scalability)، ومدى استفادته من قدرات العتاد المختلفة، سواء على مستوى المعالجات المركزية أو الرسومية.

كما تهدف هذه النتائج إلى توضيح الفروقات العملية بين بيئتي التنفيذ، وإبراز الدور الذي تلعبه المعالجة المتوازية في تحسين أداء أنظمة معالجة اللغة الطبيعية عند التعامل مع كميات كبيرة من البيانات النصية.

جدول 2. جدول مقارنة الأداء بين CPU و GPU (CUDA) في حالتي المعالجة التسلسلية والمعالجة المتوازية

عدد النصوص	بيئةCPU			بيئةGPU			
	الزمن		نسبة التحسن (%)	الزمن		نسبة التحسن (%)	GPU Memory (MB)
	التسلسلي (ث)	المتوازي (ث)		التسلسلي (ث)	المتوازي (ث)		
100	1520.55	1095.09	27.98	530.84	350.43	33.99	5222.11
500	2749.10	2027.44	26.25	2176.11	1846.83	15.13	5527.88
1000	5918.98	3905.92	34.03	5629.12	3583.82	36.33	5605.91

تُظهر النتائج التجريبية المقارنة بين بيئتي CPU و GPU (CUDA) أن أسلوب التنفيذ المتوازي يحقق تحسنًا ملحوظًا في زمن المعالجة عند جميع أحجام البيانات المختبرة (100، 500، 1000 نص). فعلى مستوى CPU تراوحت نسب التحسن بين 27.98% و34.03%، مع أداء أفضل نسبيًا عند الأحجام الكبيرة، مما يعكس استفادة تدريجية من التوازي رغم محدودية الموارد العتادية.

في المقابل، أظهرت بيئة GPU أداءً أعلى من حيث التسريع في بعض الحالات، حيث بلغت نسبة التحسن 33.99%عند 100 نص و 36.33% عند 1000 نص، بينما انخفضت إلى 15.13% عند 500 نص، وهو ما يمكن تفسيره بتكاليف التهيئة (overhead) وإدارة الدُفعات (batch processing) التي تؤثر على الكفاءة عند الأحجام المتوسطة.

كما تُبين النتائج أن استهلاك الذاكرة في GPU ظل مستقرًا نسبيًا مع زيادة حجم البيانات، مما يدل على كفاءة إدارة الموارد الرسومية ضمن النظام المقترح. وبشكل عام، تؤكد النتائج

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

أن استخدام GPU مع التوازي القائم على الدُفعات يوفر أداءً أفضل في السيناريوهات ذات الحجم الكبير، بينما يظل CPU فعالًا في المعالجة المتوازنة للبيانات الصغيرة والمتوسطة، مما يعكس قابلية النظام للتكيف مع بيئات تشغيل مختلفة.

والجداول التالية توضح أزمنة بعض العينات لبعض النصوص العربية والانجليزية ودرجة الثقة لتحديد التصنيف النصوص ، وكانت نسبة التطابق بين نتائج المعالجة التسلسلية والمتوازية 100%، مما يؤكد ثبات النموذج وعدم تأثر النتائج بطريقة التنفيذ.

جدول 3. أزمنة بعض العينات لبعض النصوص العربية والانجليزية ودرجة الثقة لتحديد التصنيف النصوص في بيئة CPU

العينات	عدد الكلمات	زمن المعالجة		درجة الثقة	
		متسلسلة	متوازية	متسلسلة	متوازية
العينة (1) "طبي عربي"	192	67.39	7.17	0.93	0.93
العينة (2) "طبي انجليزي"	194	102.53	3.69	0.74	0.74
العينة (3) "قانوني عربي"	568	212.53	22.04	0.69	0.69
العينة (4) "قانوني انجليزي"	772	411.69	25.09	0.75	0.75

جدول 4. أزمنة بعض العينات لبعض النصوص العربية والانجليزية ودرجة الثقة لتحديد التصنيف النصوص في بيئة GPU

العينات	عدد الكلمات	زمن المعالجة		درجة الثقة	
		متسلسلة	متوازية	متسلسلة	متوازية
العينة (1) "طبي عربي"	192	35.24	1.79	0.93	0.93
العينة (2) "طبي انجليزي"	194	86.74	4.5	0.75	0.75
العينة (3) "قانوني عربي"	568	126.71	8.13	0.69	0.69
العينة (4) "قانوني انجليزي"	772	319.28	25.45	0.75	0.75

ولتحليل مستوى استخدام وحدة معالجة الرسوميات (GPU) واستهلاك الذاكرة ومعدل الإنتاجية (Throughput). أظهرت النتائج أن استخدام وحدة معالجة الرسوميات (GPU) ساهم بشكل واضح في تحسين أداء النظام، خاصة عند زيادة حجم الدفعة (Batch

http://www.doi.org/10.62341/istj-vol38-2-ka14

Size). فعند استخدام حجم دفعة قدره (64)، بلغ متوسط استهلاك الذاكرة حوالي (4932.42 MB) من إجمالي (8188 MB)، أي ما يعادل تقريبًا (60.24%) من سعة الذاكرة، وهو استهلاك متوسط يدل على عدم استغلال كامل لموارد GPU.

ومع زيادة حجم الدفعة إلى (256)، بقي استهلاك الذاكرة في مستوى مقارب حيث بلغ (4908.25 MB)، أي حوالي (59.94%) من إجمالي الذاكرة، مما يشير إلى كفاءة إدارة الذاكرة وإمكانية معالجة عدد أكبر من العينات دون زيادة كبيرة في الاستهلاك. وفي المقابل، انعكس ذلك إيجابيًا على الأداء، حيث ارتفع معدل الإنتاجية للنصوص العربية من (14.65 سطر/ثانية) إلى (16.99 سطر/ثانية)، مما يؤكد أن زيادة حجم الدفعة تؤدي إلى تحسين الاستفادة من قدرات المعالجة المتوازية في GPU.

أما بالنسبة للنصوص الإنجليزية، فقد ظل معدل الإنتاجية منخفضًا نسبيًا حيث بلغ (0.90 سطر/ثانية) عند Batch=64 و(0.98 سطر/ثانية) عند Batch=256، وذلك بسبب اعتماد نموذج التصحيح الإنجليزي على المعالجة التسلسلية باستخدام CPU، مما جعله يمثل عنق زجاجة (Bottleneck) في النظام ويحد من الأداء العام.

وعلى مستوى الأداء الكلي، أظهرت النتائج اختلافًا واضحًا بين حالتي Batch Size. ففي حالة Batch=64، لم تحقق المعالجة المتوازية أي تحسن يُذكر، حيث بلغ معامل التسريع (Speedup) (0.87)، مما يشير إلى أن التنفيذ المتوازي كان أقل كفاءة من التنفيذ التسلسلي.

في المقابل، ومع زيادة Batch Size إلى (256) وكان عدد النصوص 60 نص عربي انجليزي، تحسن الأداء بشكل ملحوظ، حيث انخفض زمن التنفيذ من (602.63 ثانية) التسلسلي إلى (509.01 ثانية) توازي، وبلغت نسبة التحسن (15.53%)، مع معامل تسريع قدره (1.184)، مما يدل على فعالية المعالجة المتوازية عند استخدام دفعات كبيرة. كما ارتفع معدل الإنتاجية الكلي من (3.25 سطر/ثانية) في التنفيذ التسلسلي إلى (4.49 سطر/ثانية) في التنفيذ المتوازي، مما يعزز من فرضية أن الأداء يتحسن بزيادة حجم الدفعة.

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

وأخيرًا، بلغت كفاءة استخدام GPU (Efficiency) حوالي (0.0024)، وهي قيمة منخفضة نسبيًا، مما يشير إلى وجود مجال كبير لتحسين استغلال الموارد، خاصة أن نسبة استخدام الذاكرة لم تتجاوز ربع السعة الكلية، مما يدل على إمكانية زيادة حجم الدفعات أو تحسين توازي العمليات لتحقيق أداء أفضل.

وقد تم دراسة تأثير حجم الدفعة (Batch Size) على الأداء حيث تم تطبيق هذه الدراسة لعدد 200 نص عربي وانجليزي بأحجام مختلفة وقد كانت النتائج المتحصل عليها كما في الجدول التالي:

جدول 5. النتائج المتحصل عليها لتأثير حجم الدفعة (Batch Size) على الأداء

حجم الدفعة (Batch Size)	استهلاك GPU (MB)	الزمن التسلسلي (ثانية)	الزمن المتوازي (ثانية)	نسبة التحسن (%)	معامل التسريع (Speedup)	إنتاجية العربية (سطر/ثانية)	إنتاجية الإنجليزية (سطر/ثانية)
8	4759.48	947.67	1001.66	−5.7	0.95	11.62	0.98
32	5367.75	911.05	762.11	16.35	1.2	15.7	1.07
64	5311.86	885.36	725.61	18.04	1.22	16.59	1.1
128	5361.3	878.76	725.16	17.48	1.21	16.7	1.11
256	5359.93	1005.99	966.25	3.95	1.04	11.48	0.94
512	5361.6	1256.04	737.4	41.29	1.7	11.74	0.97
1024	5361.44	867.08	731.51	15.64	1.19	16.74	1.13

حيث أظهرت نتائج الدراسة أن حجم الدفعة (Batch Size) له تأثير مباشر وواضح على أداء النظام من حيث زمن التنفيذ، معامل التسريع، معدل الإنتاجية، واستهلاك ذاكرة GPU.فقد بيّنت النتائج أن الأحجام الصغيرة جدًا مثل 8 لم تستغل قدرات المعالج الرسومي بالشكل الأمثل، حيث سجلت أبطأ زمن في التنفيذ المتوازي بلغ 1001.66 ثانية مع معامل تسريع أقل من الواحد الصحيح (0.95) مما يدل على أن التنفيذ المتوازي كان أبطأ من التسلسلي في هذه الحالة.

في المقابل، حققت الأحجام المتوسطة مثل 64 و128 أفضل أداء عام، إذ سجلت أقل زمن تنفيذ متوازي بلغ 725.61 ثانية عند حجم دفعة 64 و725.16 ثانية عند حجم

http://www.doi.org/10.62341/istj-vol38-2-ka14

دفعة 128، مع معامل تسريع تجاوز 1.20، مما يشير إلى الاستفادة المثلى من إمكانيات GPU وتحقيق توازن جيد بين السرعة والإنتاجية.

أما الأحجام الكبيرة جدًا مثل 256 فقد أدت إلى تراجع الأداء، حيث ارتفع زمن التنفيذ وانخفض معدل الإنتاجية، ويُعزى ذلك إلى زيادة الحمل على الذاكرة وارتفاع تكلفة الحشو (Padding) واختلاف أطوال النصوص داخل الدفعة الواحدة.

ومن جهة أخرى، حقق الحجم 1024أعلى معدل إنتاجية للنصوص العربية بلغ 16.74 سطر/ثانية، إلا أن زمن التنفيذ لم يكن الأفضل، مما يدل على أنه مناسب في حالات التركيز على الإنتاجية العالية أكثر من تقليل الزمن الكلي.

كما أظهرت النتائج أن استهلاك ذاكرة GPU ارتفع مع زيادة حجم الدفعة من 4759MB عند حجم 8 إلى ما يقارب 5360MB عند الأحجام الأكبر، ثم استقر تقريبًا بعد ذلك، مما يدل على وصول الذاكرة إلى حد الاستغلال الأقصى.

وبناءً على النتائج، يُستنتج أن أفضل إعداد عملي للنظام هو استخدام Batch Size بين 64 و128، حيث يحقق توازنًا مثاليًا بين زمن التنفيذ وكفاءة استخدام الموارد الحاسوبية.

المناقشة:

أظهرت النتائج التجريبية أن النظام المقترح حقق توازنًا واضحًا بين جودة المعالجة اللغوية وكفاءة الأداء الحسابي، وهو ما يعكس نجاح التصميم الهجين القائم على دمج نماذج تعلم عميق متخصصة مع أساليب تنفيذ متوازية على بيئات عتادية مختلفة. ويُعد هذا التوازن من المتطلبات الأساسية في أنظمة معالجة اللغة الطبيعية الحديثة، إذ إن رفع جودة التصحيح دون مراعاة زمن التنفيذ يحد من الاستخدام العملي، كما أن تسريع المعالجة دون الحفاظ على الدقة يقلل من موثوقية النظام.

في جانب الجودة اللغوية، بينت نتائج مؤشري BLEU و GLEU تفوق النظام المقترح على النظام المرجعي المعتمد على أداة LanguageTool، ولا سيما في النصوص العربية، حيث سجل متوسط BLEU قيمة (0.7454) مقابل (0.6871) للنظام المرجعي، كما ارتفع GLEU إلى (0.7529) مقابل (0.6980). ويُعزى هذا التحسن إلى قدرة النماذج العميقة المستخدمة على تمثيل العلاقات السياقية بعيدة المدى، وفهم

http://www.doi.org/10.62341/istj-vol38-2-ka14

البنية الصرفية والتركيبية للغة العربية، وهي خصائص يصعب على الأنظمة التقليدية المعتمدة على القواعد الثابتة التعامل معها بالكفاءة نفسها. فاللغة العربية تتسم بغنى صرفي، وتعدد في الصيغ الاشتقاقية، ومرونة تركيبية، مما يجعل النماذج السياقية أكثر قدرة على اختيار التصحيح الأنسب وفق السياق.

أما في النصوص الإنجليزية، فقد كان التحسن محدودًا نسبيًا، حيث حافظ النظام المقترح على أداء مرتفع مقارنة بالنظام المرجعي. ويمكن تفسير ذلك بأن أدوات التصحيح الإنجليزية التقليدية وصلت أصلًا إلى مستوى نضج جيد بسبب وفرة الموارد اللغوية والقواعد النحوية المتاحة لهذه اللغة، مما يقلل هامش التحسن الممكن مقارنة باللغة العربية. ومع ذلك، فإن حفاظ النظام المقترح على تفوقه وإن كان طفيفًا يدل على استقراره وقدرته على العمل بكفاءة عبر لغتين مختلفتين من حيث البنية اللغوية.

وتؤكد عينات النصوص المعروضة قبل التصحيح وبعده أن التحسين لم يقتصر على إصلاح الأخطاء الإملائية فقط، بل شمل إعادة بناء الجمل، وتحسين علامات الترقيم، وتصحيح التراكيب النحوية، والحفاظ على المعنى العام للنص. وهذا يشير إلى أن النظام لا يعمل كمصحح سطحي للكلمات، بل كنظام إعادة صياغة لغوية موجهة بالسياق، وهي ميزة مهمة في التطبيقات الاحترافية مثل النصوص الطبية والقانونية التي تتطلب دقة عالية.

أما من ناحية أداء الاستدلال، فقد أثبتت النتائج أن المعالجة المتوازية حسّنت زمن التنفيذ في جميع البيئات المختبرة مقارنة بالمعالجة التسلسلية. ففي بيئة CPU تراوحت نسب التحسن بين (27.98%) و(34.03%)، بينما سجلت بيئة GPU نسبًا تراوحت بين (15.13%) و(36.33%) تبعًا لحجم البيانات. ويشير ذلك إلى أن الاستفادة من التوازي ترتبط بحجم عبء العمل؛ فكلما زاد عدد النصوص ارتفعت كفاءة توزيع العمليات وتقاسم الموارد، وهو ما يتفق مع مبادئ الحوسبة المتوازية.

ومن الناحية النظرية، يمكن تفسير هذا السلوك وفق قانون Amdahl، الذي ينص على أن مقدار التسريع الكلي يظل مقيدًا بنسبة الأجزاء غير القابلة للتوازي داخل البرنامج. لذلك، فإن وجود مراحل تسلسلية مثل قراءة الملفات، والمعالجة المسبقة، وتحميل النماذج،

http://www.doi.org/10.62341/istj-vol38-2-ka14

ونقل البيانات بين الذاكرة الرئيسية وذاكرة GPU، يمنع الوصول إلى تسريع خطي كامل، خاصة عند أحجام البيانات الصغيرة أو المتوسطة. ولهذا ظهرت بعض الفروقات المحدودة في نسب التحسن بين التجارب المختلفة.

كما أوضحت النتائج أن تفوق GPU لا يعتمد فقط على توفر العتاد الرسومي، بل يرتبط أيضًا بكيفية تنظيم البيانات داخل الدفعات (Batching) وتقليل كلفة النقل بين الذاكرة والمعالج الرسومي. فقد أظهرت الدراسة أن أفضل أداء عملي تحقق عند أحجام دفعة متوسطة، خاصة بين 64و128، حيث سُجلت أقل أزمنة تنفيذ مع أفضل توازن بين السرعة واستهلاك الموارد. ويُفسر ذلك بأن الأحجام الصغيرة جدًا لا تستغل العدد الكبير من الأنوية الحسابية في GPU بصورة كافية، بينما قد تؤدي الأحجام الكبيرة جدًا إلى زيادة عمليات الحشو (Padding) وارتفاع الحمل على الذاكرة.

ويرتبط ذلك أيضًا بمفهوم موازنة الحمل (Load Balancing)، إذ إن نجاح التنفيذ المتوازي يعتمد على توزيع العمل بصورة متجانسة بين الأنوية الحسابية. وعندما تختلف أطوال النصوص داخل الدفعة الواحدة، تضطر بعض الخيوط الحسابية إلى الانتظار حتى انتهاء خيوط أخرى، مما يؤدي إلى انخفاض الكفاءة الزمنية. لذلك ظهرت الأحجام المتوسطة كحل عملي يوازن بين كثافة التشغيل وكلفة التنظيم الداخلي للدفعات.

ومن الناحية المعمارية، فإن نماذج Transformer المستخدمة في النظام تعتمد بدرجة كبيرة على عمليات ضرب المصفوفات وآليات الانتباه الذاتي (Self–Attention)، وهي عمليات مناسبة بطبيعتها للتنفيذ المتوازي على GPU. لذلك فإن التحسن الزمني الملحوظ لا يمثل مجرد تحسين برمجي، بل يعكس توافقًا مباشرًا بين بنية النموذج الرياضي وبنية المعالج الرسومي، وهو عامل حاسم في تصميم الأنظمة الذكية عالية الأداء.

وأظهرت قياسات الذاكرة أن استهلاك GPU ظل مستقرًا نسبيًا ضمن حدود تقارب 5.3GB عبر معظم التجارب، مما يشير إلى كفاءة إدارة الموارد وإمكانية توسيع النظام لمعالجة أحجام بيانات أكبر مستقبلًا دون الحاجة إلى زيادة كبيرة في المتطلبات العتادية. كما يدل هذا الاستقرار على أن عنق الزجاجة الرئيس في بعض الحالات لم يكن الذاكرة، بل طريقة جدولة المهام وأحجام الدفعات المختارة.

http://www.doi.org/10.62341/istj-vol38-2-ka14

ومن النتائج المهمة كذلك أن نسبة التطابق بين مخرجات التنفيذ التسلسلي والمتوازي بلغت 100%، مما يؤكد أن تسريع التنفيذ لم يؤثر في دقة التصنيف أو جودة التصحيح، بل اقتصر أثره على تقليل زمن المعالجة فقط. وتعد هذه النتيجة مهمة في التطبيقات الواقعية، لأن بعض تقنيات التسريع قد تُحدث فروقات طفيفة في النتائج، وهو ما لم يظهر في النظام المقترح.

كما أظهرت درجات الثقة الخاصة بمهام التصنيف استقرارًا واضحًا في مختلف بيئات التشغيل، مما يعني أن تغيير أسلوب التنفيذ أو منصة المعالجة لم يؤثر في قرارات النموذج التصنيفية. ويعكس ذلك قوة التعميم وقابلية النظام للنشر على منصات تشغيل متنوعة دون فقد في الاعتمادية.

وتشير النتائج بصورة أعمق إلى أن نجاح أنظمة معالجة اللغة الطبيعية لا يعتمد على جودة النموذج فقط، بل على تكامل ثلاثي بين الخوارزمية، وإطار التنفيذ البرمجي، والعتاد الحاسوبي .فحتى وجود بطاقة رسومية قوية لا يضمن أفضل أداء ما لم تكن الخوارزميات مصممة للاستفادة من التوازي، وما لم يتم تنظيم البيانات بكفاءة. وهذه النتيجة تمثل مبدأ أساسيًا في تصميم الأنظمة الذكية الحديثة.

وبصورة عامة، تؤكد النتائج أن دمج نماذج اللغة العميقة مع تقنيات التوازي الحاسوبي يمثل اتجاهًا فعالًا لتطوير أنظمة معالجة نصوص ثنائية اللغة تجمع بين الجودة العالية والكفاءة الزمنية، مع قابلية جيدة للتوسع في التطبيقات الواقعية مثل أنظمة التدقيق اللغوي، الأرشفة الذكية، معالجة المستندات المؤسسية، ومحركات تحليل المحتوى متعددة اللغات.

ورغم النتائج الإيجابية، فإن الدراسة تكشف أيضًا عن فرص تطوير مستقبلية، من أهمها توسيع نطاق الاختبار ليشمل مجالات نصية إضافية، واستخدام نماذج أخف وزنًا لخفض زمن الاستدلال، وتطبيق أساليب تحسين مثل Quantization وONNX Runtime، بالإضافة إلى دراسة الأداء على منصات سحابية متعددة البطاقات الرسومية. ومن المتوقع أن تسهم هذه الاتجاهات في رفع كفاءة النظام وزيادة جاهزيته للاستخدام الصناعي واسع النطاق.

تقييم أداء ودقة نماذج التصحيح والتصنيف اللغوي ثنائية اللغة (عربي–إنجليزي) على بيئات CPU و GPU باستخدام التنفيذ التسلسلي والمتوازي

http://www.doi.org/10.62341/istj-vol38-2-ka14

الخلاصة

قدّمت هذه الدراسة نظامًا مقترحًا لمعالجة النصوص العربية والإنجليزية يجمع بين التصحيح اللغوي، تصنيف النصوص، والتنفيذ المتوازي على بيئتي CPU و GPU. وأظهرت النتائج أن النظام حقق أداءً لغويًا متقدمًا مقارنة بالنظام المرجعي، خاصة في اللغة العربية، حيث سجل تحسنًا واضحًا في مؤشري BLEU و GLEU.

كما أثبتت التجارب أن استخدام المعالجة المتوازية أدى إلى تقليل زمن التنفيذ بصورة ملحوظة في مختلف أحجام البيانات، مع أفضلية واضحة لبيئة GPU عند الأحجام الكبيرة. وبينت الدراسة أن اختيار حجم الدفعة المناسب يُعد عاملًا حاسمًا في تحقيق أفضل أداء، حيث وفرت القيم بين 64 و128 أفضل توازن بين السرعة واستهلاك الموارد. وأظهرت النتائج كذلك ثبات مخرجات النظام بين التنفيذ التسلسلي والمتوازي بنسبة تطابق كاملة، مما يعزز موثوقية النظام وقابليته للاستخدام العملي.

وبناءً على ذلك، يمكن الاستنتاج أن النظام المقترح يمثل إطارًا فعالًا لمعالجة النصوص ثنائية اللغة، يجمع بين جودة التصحيح وسرعة التنفيذ، ويُعد مناسبًا للتطبيقات التي تتطلب معالجة كميات كبيرة من البيانات النصية في الزمن شبه الحقيقي.

أما مستقبلًا، فيمكن تطوير النظام من خلال توحيد جميع مراحل المعالجة ضمن بيئة GPU، وتحسين النماذج الإنجليزية، وتوسيع الدعم ليشمل مجالات لغوية إضافية ونصوصًا أكثر تعقيدًا.

References

[1] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer,” *Journal of Machine Learning Research (JMLR)*, vol. 21, no. 140, pp. 1–67, 2020, doi: 10.5555/3455716.3455856.

[2] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov,
“XLM-R: Unsupervised Cross-lingual Representation Learning at Scale,” in Proc. 58th Annual Meeting of the Association for

http://www.doi.org/10.62341/istj-vol38-2-ka14

Computational Linguistics (ACL), Online, Jul. 2020, pp. 8440–8451. doi: 10.18653/v1/2020.acl-main.747.

[3] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, et al., "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," *arXiv preprint arXiv:1609.08144*, 2016.

[4] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The Long-Document Transformer," *arXiv preprint arXiv:2004.05150*, 2020.

[5] J. Ghorpade, J. Parande, M. Kulkarni, and A. Bawaskar, "GPGPU Processing in CUDA Architecture," , Feb. 2012. [Online]. Available: https://arxiv.org/abs/1202.4347.

[6] O. Obeid, N. Zalmout, S. Khalifa, D. Taji, M. Oudah, B. Alhafni, G. Inoue, F. Eryani, A. Erdmann, and N. Habash, "CAMeL Tools: An Open Source Python Toolkit for Arabic Natural Language Processing," in *Proc. 12th International Conference on Language Resources and Evaluation (LREC)*, Marseille, France, May 2020, pp. 7022–7032/ Available: https://aclanthology.org/2020.lrec-1.868.

[7] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, et al., "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020, doi:10.48550/arxiv.1910.10683.

[8] J. Zhu, X. Shi, S. Zhang, R. Ali, "Machine Learning-Based Grammar Error Detection Method in English Composition," *Scientific Programming*, vol. 2021, pp. 1–10, Dec. 2021, doi: 10.1155/2021/4213791.

[9] P. He, X. Liu, J. Gao, and W. Chen, "DeBERTa: Decoding-Enhanced BERT with Disentangled Attention," arXiv preprint arXiv:2006.03654, Jun. 2020. [Online]. Available: https: //arxiv.org/abs/2006.03654.

[10] F. C. Coelho, R. R. Souza, Á. Justen, F. Amieiro, and H. Mello, "PyPLN: a Distributed Platform for Natural Language Processing," arXiv preprint arXiv:1301.7738, Jan. 2013. [Online]. Available: https://arxiv.org/abs/1301.7738.

[11] A. Argueta and D. Chiang, "Decoding with Finite-State Transducers on GPUs," *arXiv preprint arXiv:1701.03038*,

http://www.doi.org/10.62341/istj-vol38-2-ka14

Jan. 2017. [Online]. Available: https://arxiv.org/abs/1701.03038.

[12] A. Argueta and D. Chiang, "Composing Finite State Transducers on GPUs," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, Melbourne, Australia, Jul. 2018, pp. 2697–2705, doi: 10.18653/v1/P18-1251. [Online]. Available: https://aclanthology.org/P18-1251.

[13] S. Narang, G. Diamos, J. Alben, P. Micikevicius, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh and H. Wu, "Mixed Precision Training," https://arxiv.org/abs/1710.03740.

[14] K. Omelianchuk, V. Atrasevych, A. Chernodub, and O. Skurzhanskyi, "GECToR – Grammatical Error Correction: Tag, Not Rewrite," in *Proc. 15th Workshop on Innovative Use of NLP for Building Educational Applications (BEA@ACL)*, 2020, pp. 163–170. [Online]. Available: https://aclanthology.org/2020.bea-1.16.

[15] B. Alhafni, G. Inoue, C. Khairallah, and N. Habash, "Advancements in Arabic Grammatical Error Detection and Correction: An Empirical Investigation," in Proc. 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP), Singapore, Dec. 2023, pp. 6430–6448, doi: 10.18653/v1/2023.emnlp-main.396. [Online]. Available: https://aclanthology.org/2023.emnlp-main.396/

[16] A. Rozovskaya, H. Bouamor, N. Habash, W. Zaghouani, O. Obeid, and B. Mohit, "The Second QALB Shared Task on Automatic Text Correction for Arabic," in *Proc. The 2nd Workshop on Arabic Natural Language Processing (ANLP)* at ACL, Beijing, China, 2015, pp. 26–35. DOI: 10.18653/v1/W15-3204.
[Online]. Available: https://aclanthology.org/W15-3204.

[17] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, et al., "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," in *Proc. ACL*, 2020, pp. 7871–7880, doi: 10.18653/v1/2020.acl-main.703.

http://www.doi.org/10.62341/istj-vol38-2-ka14

[18] FCE Dataset: First Certificate in English (ESOL) corpus, Cambridge Learner Corpus subset, 2011. [Online]. Available: https://paperswithcode.com/dataset/fce

[19] J. Ng, J. Tetreault, and C. Brockett, "The CoNLL-2014 Shared Task on Grammatical Error Correction," in *Proc. ACL*, 2014, pp. 1–14, doi: 10.3115/v1/W14-1701.

[20] J. Yin, Y. Liu, and A. Schütze, "Benchmarking Zero-Shot Text Classification: Datasets, Evaluation and Entailment Approach," in *Proc. EMNLP*, 2019, pp. 3917–3926, doi: 10.18653/v1/D19-1404.

[21] A. Williams, N. Nangia, and S. R. Bowman, "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference," in Proc. 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), New Orleans, LA, 2018, pp. 1112–1122, doi: 10.18653/v1/N18-1101. [Online]. Available: https://aclanthology.org/N18-1101/.

[22] A. Conneau, R. Rinott, G. Lample, A. Williams, S. Bowman, H. Schwenk, and V. Stoyanov, "XNLI: Evaluating Cross-lingual Sentence Representations," in *Proc. 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Brussels, Belgium, 2018, pp. 2475–2485, doi: 10.18653/v1/D18-1269. [Online]. Available: https://aclanthology.org/D18-1269.

[23] NVIDIA Corporation, CUDA C Programming Guide, 2023. [Online]. Available: https://docs.nvidia.com/cuda/cuda-programming-guide/index.html