مجلة جامعة البعث

سلسلة العلوم الهندسية الميكانيكية والكهربائية والمعلوماتية

مجلة علمية محكمة دورية

المجلد 43 . العدد 29

1442 هـ - 2021 م

تهدف المجلة إلى نشر البحوث العلمية الأصيلة، ويمكن للراغبين في طلبها الاتصال بالعنوان التالي:

رئيس تحرير مجلة جامعة البعث

سورية . حمص . جامعة البعث . الإدارة المركزية . ص . ب (77)

. هاتف / فاكس : 2138071 31 963 ++

. موقع الإنترنت : www.albaath-univ.edu.sy

. البريد الالكتروني : magazine@ albaath-univ.edu.sy

ISSN: 1022-467X

قيمة العدد الواحد : 100 ل.س داخل القطر العربي السوري

25 دولاراً أمريكياً خارج القطر العربي السوري

قيمة الاشتراك السنوي : 1000 ل.س للعموم

500 ل.س لأعضاء الهيئة التدريسية والطلاب

250 دولاراً أمريكياً خارج القطر العربي السوري

توجه الطلبات الخاصة بالاشتراك في المجلة إلى العنوان المبين أعلاه.

يرسل المبلغ المطلوب من خارج القطر بالدولارات الأمريكية بموجب شيكات باسم جامعة البعث.

تضاف نسبة 50% إذا كان الاشتراك أكثر من نسخة.

شروط النشر في مجلة جامعة البعث

الأوراق المطلوبة:

- 2 نسخة ورقية من البحث بدون اسم الباحث / الكلية / الجامعة) + CD / word من البحث منسق حسب شروط المجلة.
- طابع بحث علمي + طابع نقابة معلمين.
- **اذا كان الباحث طالب دراسات عليا:**

يجب إرفاق قرار تسجيل الدكتوراه / ماجستير + كتاب من الدكتور المشرف بموافقته على النشر في المجلة.

- **اذا كان الباحث عضو هيئة تدريسية:**

يجب إرفاق قرار المجلس المختص بإنجاز البحث أو قرار قسم بالموافقة على اعتماده حسب الحال.

- **اذا كان الباحث عضو هيئة تدريسية من خارج جامعة البعث :**

يجب إحضار كتاب من عمادة كليته تثبت أنه عضو بالهيئة التدريسية و على رأس عمله حتى تاريخه.

- **اذا كان الباحث عضواً في الهيئة الفنية :**

يجب إرفاق كتاب يحدد فيه مكان و زمان إجراء البحث ، وما يثبت صفته وأنه على رأس عمله.

- **يتم ترتيب البحث على النحو الآتي بالنسبة لكليات (العلوم الطبية والهندسية والأساسية والتطبيقية):**

عنوان البحث . . ملخص عربي و إنكليزي (كلمات مفتاحية في نهاية الملخصين).

1- مقدمة

2- هدف البحث

3- مواد وطرق البحث

4- النتائج ومناقشتها .

5- الاستنتاجات والتوصيات .

6- المراجع.

- **يتم ترتيب البحث على النحو الآتي بالنسبة لكليات (الآداب – الاقتصاد– التربية – الحقوق – السياحة – التربية الموسيقية وجميع العلوم الإنسانية):**
- عنوان البحث . . ملخص عربي و إنكليزي (كلمات مفتاحية في نهاية الملخصين).

1. مقدمة.
2. مشكلة البحث وأهميته والجديد فيه.
3. أهداف البحث و أسئلته.
4. فرضيات البحث و حدوده.
5. مصطلحات البحث و تعريفاته الإجرائية.
6. الإطار النظري و الدراسات السابقة.
7. منهج البحث و إجراءاته.
8. عرض البحث و المناقشة والتحليل
9. نتائج البحث.
10. مقترحات البحث إن وجدت.
11. قائمة المصادر والمراجع.

7- يجب اعتماد الإعدادات الآتية أثناء طباعة البحث على الكمبيوتر:

أ- قياس الورق 17.5×25 B5.

ب- هوامش الصفحة: أعلى 2.54- أسفل 2.54 – يمين 2.5- يسار 2.5 سم

ت- رأس الصفحة 1.6 / تذييل الصفحة 1.8

ث- نوع الخط وقياسه: العنوان . Monotype Koufi قياس 20

. كتابة النص Simplified Arabic قياس 13 عادي . العناوين الفرعية Simplified Arabic قياس 13 عريض.

ج . يجب مراعاة أن يكون قياس الصور والجداول المدرجة في البحث لا يتعدى 12سم.

8- في حال عدم إجراء البحث وفقاً لما ورد أعلاه من إشارات فإن البحث سيهمل ولا يرد البحث إلى صاحبه.

9- تقديم أي بحث للنشر في المجلة يدل ضمناً على عدم نشره في أي مكان آخر، وفي حال قبول البحث للنشر في مجلة جامعة البعث يجب عدم نشره في أي مجلة أخرى.

10- الناشر غير مسؤول عن محتوى ما ينشر من مادة الموضوعات التي تنشر في المجلة

11- تكتب المراجع ضمن النص على الشكل التالي: [1] ثم رقم الصفحة ويفضل استخدام التهميش الإلكتروني المعمول به في نظام وورد WORD حيث يشير الرقم إلى رقم المرجع الوارد في قائمة المراجع.

تكتب جميع المراجع باللغة الانكليزية (الأحرف الرومانية) وفق التالي:

آ . إذا كان المرجع أجنبياً:

الكنية بالأحرف الكبيرة . الحرف الأول من الاسم تتبعه فاصلة . سنة النشر . وتتبعها معترضة (–) عنوان الكتاب ويوضع تحته خط وتتبعه نقطة . دار النشر وتتبعها فاصلة . الطبعة (ثانية . ثالثة) . بلد النشر وتتبعها فاصلة . عدد صفحات الكتاب وتتبعها نقطة.

وفيما يلي مثال على ذلك:

–MAVRODEANUS, R1986– **Flame Spectroscopy**. Willy, New York, 373p.

ب . إذا كان المرجع بحثاً منشوراً في مجلة باللغة الأجنبية:

. بعد الكنية والاسم وسنة النشر يضاف عنوان البحث وتتبعه فاصلة، اسم المجلد ويوضع تحته خط وتتبعه فاصلة . المجلد والعدد (كتابة مختزلة) وبعدها فاصلة . أرقام الصفحات الخاصة بالبحث ضمن المجلة.

مثال على ذلك:

BUSSE,E 1980 Organic Brain Diseases **Clinical Psychiatry News** , Vol. 4. 20 – 60

ج. إذا كان المرجع أو البحث منشوراً باللغة العربية فيجب تحويله إلى اللغة الإنكليزية و التقيد

بالبنود (أ و ب) ويكتب في نهاية المراجع العربية: (المراجع In Arabic)

رسوم النشر في مجلة جامعة البعث

1. **دفع رسم نشر (20000) ل.س عشرون ألف ليرة سورية عن كل بحث لكل باحث يريد نشره في مجلة جامعة البعث.**
2. **دفع رسم نشر (50000) ل.س خمسون الف ليرة سورية عن كل بحث للباحثين من الجامعة الخاصة والافتراضية .**
3. **دفع رسم نشر (200) مئتا دولار أمريكي فقط للباحثين من خارج القطر العربي السوري .**
4. **دفع مبلغ (3000) ل.س ثلاثة آلاف ليرة سورية رسم موافقة على النشر من كافة الباحثين.**

المحتوى

معالجة صور الأقمار الصّناعيّة باستخدام شبكات التّعلّم العميق على منصّات البيانات الكبيرة

م. عروه احمد قصاب د.م. أحمد محمود أحمد د.م. عهد نصر البودي

كلية الهندسة المعلوماتية – جامعة تشرين

الملخص

مكّن التقدّم الكبير الحاصل في تقنيات الاستشعار عن بعد من جعل حسّاسات الأقمار الصّناعيّة قادرة على التقاط صور عالية الوضوح والدّقّة وذات حزم طيفيّة متعدّدة. تحوي هذه الصّور معلومات قيّمة يمكن استخدامها في تطبيقات معالجة الصّور في عدّة مجالات مختلفة، كتصنيف الغطاء اللأرضي واكتشاف التغيّرات والتحذير المبكّر من الكوارث. الّا أنّ هذه التّطبيقات وخصوصاً تلك التي تعتمد على تقنيات التّعلّم العميق تواجه مشكلة الحاجة الكبيرة لموارد الحوسبة أثناء معالجة بياناتها إضافةً الى الوقت الكبير المُستهلك عند تنفيذها على النّظم التّقليديّة، وهذه المشكلة تزداد سوءا للتطبيقات التي تتطلب سرعة استجابة عالية. الأكثر من ذلك وبسبب الازدياد الكبير في الصّور المُولّدة عن طريق الاقمار الصّنعيّة أصبحت هناك حاجة لوجود أنظمة قادرة على استيعاب وتخزين حجوم ضخمة من الصّور المولّدة وبنفس الوقت لها القدرة على التّوسّع عند زيادة حجمها. يدرس هذا البحث استخدام النّظم فائقة الأداء ومنصّات البيانات الكبيرة باستخدام سبارك وهادووب واستثمارها في معالجة صور الأقمار الصّناعيّة بمساعدة أُطر عمل تفرّعيّة مُخصّصّة (Horovod, TensorFlowOnSpark)، حيث تمّ العمل على نمّوذج تعلّم عميق للتّقطيع الدّلاليّ (U-Net) كمثالٍ عمليٍّ وقُيّم أداءُ النّظام في كلّ من برنامجي التّدريب والتّنبؤ. هذا وقد أظهرت النّتائج تحسيناً كبيراً في زمن التّنفيذ للمنهجيّة

المُقترحة بالمقارنة مع منهجيّة الجهاز المنفرد وقدرةً على التّوسّع لحجوم مختلفة من عناقيد البيانات الكبيرة.

الكلمات المفتاحيّة

التّعلّم العميق – معالجة صور الأقمار الصّناعيّة – نمّوذج U-Net للتّقطيع الدّلاليّ – المُعالجة التّفرّعيّة – النّظم فائقة الأداء – منصّات البيانات الكبيرة

Satellite Image Processing Using Deep Learning Networks on Big Data Platforms

Eng. Orwa Ahmad Kassab Dr. Ahmad Mahmoud Ahmad Dr. Ahed Alboody

Faculty of Informatics Engineering, Tishreen University

Abstract

The recent advancement in remote sensing technologies made satellite sensors capture high-resolution and hyper-spectral images. These images contain valuable information and can be utilized in many applications and different disciplines like land cover classification, change detection and early warning of disasters. These applications especially that depend on deep learning technologies face the problem of high demands of computation resources and take much time to execute on traditional systems, and this problem got worse for applications that require rapid response. Moreover, the increasing amount of images generated by satellites needs a system that can store massive volumes of data and has the ability to scale when their size increases. This paper investigates the use of high-performance systems and big data platforms where Spark and Hadoop can be utilized in processing images with the help of specialized distributed computing frameworks like Horovod and TensorFlowOnSpark. A segmentation deep learning model

was taken as an example application (U-Net) and the system performance was evaluated in training and prediction programs. Results showed a significant improvement in execution time of the proposed approach compared to the single machine approach and an ability to scale for different big data cluster sizes.

Key words: Deep Learning, Satellite Image Processing, U-Net Semantic Segmentation, Distributed Processing, High-Performance Systems, Big Data Platforms,

1- مقدّمة

تتوافر اليوم العديد من الشركات العاملة في مجال الاستشعار عن بعد والتي تؤمّن صوراً مُلتقطة عن طريق أقمار صنعيّة وتوظفها في العديد من المجالات المختلفة كاكتشاف التغيرات وتصنيف المناطق والإنذار المبكّر من الكوارث وجمع احصائيّات عن مناطق واسعة من الأرض كالمحاصيل او توزّع الغابات والجفاف وغيرها. ومع دخول عصر المراقبة الأرضية عالية الدقة أصبحت بيانات الاستشعار عن بعد ذات حجوم كبيرة ومتوفّرة بشكل يومي وتمرّ بمرحلة تضخم هائل، ويُوصف نُموّها على أنّه انفجاريّ، كما انّ انتشار هذا النّوع من البيانات يدفع بقوّة نحو زيادة التعقيد في التّعامل معها نتيجة تنوّعها وتعدّد خصائصها وأبعادها [1]. وقد وصلت مرحلة وضوح الصور المقدّمة درجة أن يصل حجمها الى ما يقارب ال 50 GB ، وهذا يُعتبر فائق الحجم بالمقارنة مع الصّور الملتقطة بالكاميرات الرّقميّة الحديثة التي تتزّود بها الأجهزة الذكيّة المنتشرة بكثرة.

يتم استثمار هذه البيانات في العديد من التّطبيقات الخدميّة المختلفة والمتنوّعة من ناحية الاغراض الّا أنّ الاستجابة الزمنية لتطبيقات معالجة الصور الفضائية وفق التقنيات الحديثة وخصوصا تلك التي تعتمد على تقنيات الذكاء الصّنعي بطيئة بسبب الزمن الكبير المستهلك للقيام بذلك نتيجة تعقيد خوارزمياتها ومتطلباتها العالية جدّا في الحوسبة، اضافة لذلك يوجد طلب متزايد على تقديم سرعات عالية تلبيةً لرغبات الزّبائن اكثر من أي وقت مضى. الاكثر من ذلك مشكلة التعامل مع هذه البيانات تزيد بشكل كبير في حال كنا نتعامل مع الفيديو الفضائي الحديث او التطبيقات التي تتطلب معالجة في الزمن الحقيقي.

إنّ استخدام الطرق والانظمة التقليدية في معالجة بيانات الصور والفيديو الكبيرة في تطبيقات ذات طبيعة مشابهة لما تمّ التّطرّق اليها سابقا يصطدم بمشكلتين رئيسيتين هما بطئ التنفيذ الناتج عن الحجم الكبير للبيانات من جهة وعدم قابلية التّوسّع من جهة أخرى. من هنا نجد الحاجة الكبيرة الى موارد وعتاد مكلف ومُخصّص لأداء هذه المهمّة

بسبب السّعة العالية والسّمات عالية الوضوح للبيانات وهذه يفرض عبئاً على البنية التحتية المُستثمرة في أثناء المعالجة [2].

وليست الكلفة هي المشكلة الوحيدة فقط في هذه الانظمة وانّما نجد أيضا الصعوبة العالية في قابلية توسعة الموارد عند الحاجة لأنّ طريقة توسّعها يعتمد المنهج العمودي المُرتكز اساساً على تحديث مواصفات الاجهزة بزيادة ذواكرها ومعالجاتها الحسابية وهذا يتطلب ايضا اعادة تشغيل الخدمات العاملة من جديد عند تعديل العتاد المُستخدم، اضافةً الى ذلك نجد مشكلة نقطة الفشل الواحدة(Single point of failure) التي تعني حدوث فشل ما أثناء تنفيذ ومعالجة التطبيق على الجهاز العامل من دون وجود بديل او اجهزة احتياطية تستأنف تنفيذ التطبيق، وهذا يجعل العمل ينهار بشكل كامل حيث يتمّ اللّجوء الى اعادة تشغيل التّطبيق مرّة أخرى أو الانتظار ريثما يتم اصلاح سبب المشكلة وبالتالي المزيد من التكاليف الاضافية.

مما سبق نرى أنّه اصبحنا بحاجة للانتقال الى استخدام نظم ذات أداء عالٍ قادرة على استيعاب الحجوم الكبيرة للبيانات ومعالجتها بما يتوافق مع حاجات المستثمرين لناحية السّرعة في ايصال النّتائج المرغوبة وخصوصاً في التطبيقات التي تتطلب استجابة زمنيّة سريعة جدّا او تطبيقات الزّمن الحقيقيّ، الأكثر من ذلك يجب أن تكون هذه النّظم قابلة للتوسّع بمعنى قادرة على التكيّف مع امكانيّة تضخّم البيانات مستقبلاً وتتكيّف الى حدّ ما مع حالات الفشل.

تُعتبر تقنيات البيانات الكبيرة ومنصّاتها من أهم الأدوات المُستخدمة في تلبية الاحتياجات السابقة من ناحية قدرتها على استيعاب التضخم الكبير في حجوم البيانات وفي استثمارها للنّظم المُوزّعة من خلال المعالجة التّفرّعيّة لتحقيق الأداء المرغوب، وعلى الرّغم من التّحديات الموجودة لتحقيق زمن استجابة جيدة بالنسبة للمستثمرين عند معالجة هذه البيانات باستخدام التّقنيات المذكورة فإنّه يمكن تجهيزها بطرق تساعد في تخفيض الزمن المستهلك أثناء تنفيذ التطبيقات بشكل كبير اذا ما تم إعدادها بالشكل المطلوب.

2- مشكلة البحث

أصبحت تطبيقات معالجة الصّور الرّقميّة المُلتقطة عبر أجهزة الاستشعار عن بعد ذات تعقيد كبير، كما أنّ البيانات المُولّدة عن طريق هذه الأجهزة تزداد من ناحية الحجم وتتنوّع أيضاً لتشمل الفيديو الفضائي اضافة للصور، وعلى الرّغم من التّطوّر الكبير في تقنيات التّخزين والحوسبة التقليديّة التي قدّمت امكانيّات واسعة من ناحية سرعة معالجة البيانات وتخزينها لا تزال هذه التقنيات تعاني من صعوبة كبيرة في تلبية حاجات المستثمرين لناحية قابلية اضافة بيانات جديدة باستمرار ومن ناحية تخفيض زمن التنفيذ ايضاً، وتحديداً في التطبيقات التي تطلب قدرات حوسبة عالية جدّاً كتقنيات الذكاء الاصطناعي التي تستثمر خوارزميات تعلّم الآلة والتعلّم العميق أو تطبيقات الزمن الحقيقي.

انّ المشاكل المذكورة آنفاً تمثل تحدّيات حقيقية للنظم والتقنيات التقليديّة و يحتّم علينا الانتقال الى أدوات وتقنيات أخرى تستطيع تلبية حاجات المستثمرين بمعالجة البيانات بسرعة كافية لايصال النتائج بزمن مقبول يتلائم مع طبيعة التطبيقات المُستثمرة و تكون قادرة ايضاً على استيعاب الحجوم الكبيرة والمتزايدة لهذه البيانات.

3- هدف البحث

تلقى ابحاث استخدام تقنيات البيانات الكبيرة ونظم الحوسبة فائقة الأداء في معالجة تطبيقات صور الأقمار الصناعية رواجا مطردا واهتماما بالغاً بحثياً وعلمياً خصوصا ان مستقبلاً كبيراً ينتظر هذه التقنية عند استخدام تقنيات التّعلّم العميق أو في حال توفّر الفيديو الفضائي.

تحقّق هذه الدّراسة في استخدام تقنية/تقنيات أو نظم ذات أداءٍ عالٍ تستطيع تنفيذ تطبيقات معالجة الصّور المُلتقطة بالأقمار الصّنعيّة بزمن يلبّي حاجات المستثمرين وتتمتّع بالقدرة على استيعاب حجوم بيانات كبيرة ومتزايدة، وفي نفس الوقت ستقيّم الدّراسة أداء عمل هذا النظام/النظم وامكانية استثماره على أرض الواقع. وقد اعتُمد تطبيق تصنيف الصّور

المُرتكز على تقنيات التعلّم العميق باستخدام الشّبكات العصبونية الالتفافيّة (CNN) على مستوى الكائن (object) كمثال عمليّ وتجريبيّ.

للبحث تطبيقات واسعة في تصنيف الغطاء الأرضي واكتشاف التغيرات ويمكن ان يكون له دور كبير في تصنيف المناطق المتضررة سكنيا وعلى مستو البيئة والمحاصيل الزّراعيّة في سوريا عند توفّر البيانات المطلوبة حيث يمكن استثمار هذه الدّراسة في الحصول على نتائج سريعة لو تمّ تطبيق منهجيّاتها على بنية مناسبة للعمل. وسوف نرى أنّ المنهجيّة المُطبّقة في هذا البحث لا تقتصر على صور الأقمار الصّناعية فحسب وانّما يمكن أن تُنجز على تطبيقات و صور ذات طبيعة مختلفة، كما ينسحب الأمر على بيانات كبيرة أخرى كالنّصوص والسّجلّات المتنوّعة.

سيكون من ضمن أهداف هذا البحث تقييم أداء عمل منصّات البيانات الكبيرة Hadoop و Spark في تطبيقات معالجة الصور المذكورة في عمليتيّ التّدريب والتنبّؤ حيث سيكون معامل زمن التّنفيذ معياراً أساسياً في مراقبة الأداء.

4- مواد وطرق البحث

استُخدم نمّوذج الU-net الشّهير في هذا البحث، وهو أحد نماذج التّقطيع الدّلالي التي تعتمد على الشّبكات العصبونيّة العميقة في تصنيف الصّور، حيث استعرض البحث توصيفاً لطبقات ومعمارية هذا النّموذج وآليّة عمله بحيث نحصل من خلاله على خرائط التّقطيع من صورة دخل عند القيام بعمليّة التّنبؤ. كما تناول البحث أهمّ منصّات البيانات الكبيرة وناقش اختيار المنهج الأفضل لمعالجة التطبيق المطلوب. ومن بين ما تمّت دراسته كان استخدام أُطر عمل تفرّعيّة مُخصّصة لأداء المعالجة التّفرّعية للبيانات على منصّات البيانات الكبيرة. أيضاً تطرّق البحث الى مجموعات البيانات المُستخدمة وطبيعتها وكيفيّة معالجتها.

مرّ هذا البحث بالمراحل التالية:

- تحديد الأداة الأفضل لمعالجة البيانات، واختيار النّموذج البرمجّي التّفرّعيّ الملائم
- اختيار وتركيب منصة بيانات كبيرة مناسبة للمعالجة التّفرعيّة لتطبيقات الصّور
- استخدام آلية للاستحواذ على هذه البيانات واجراء معالجة أوّلية لتوليد مجموعة جديدة وكبيرة من الصّور الجاهزة للمعالجة الفعليّة في التّطبيق
- استخدام النّظام المُقترح لتصنيف صور الأقمار الصّنعيّة على بيانات منطقة جغرافيّة مُحدّدة متاحة بشكل مجّاني باستخدام منصّتي البيانات الكبيرة هادووب وسبارك
- مقارنة وتحليل النتائج التي تمّ الحصول عليها باستخدام النّظام المُقترح

4-1- الدّراسة المرجعيّة

في الآونة الأخيرة، وُجدت عدّة دراسات تناولت كيفيّة استخدام النّظم فائقة الأداء في معالجة صور الأقمار الصّناعيّة باستخدام منصّات وتقنيات البيانات الكبيرة من خلال استثمار إمكانيّات أجهزة متعدّدة واستثمار كل من الحوسبة المُوزّعة وقدرات التّخزين لكل منها في تنفيذ التّطبيقات المختلفة بسبب قدرتها العالية على استيعاب الكمّيات الكبيرة والمتزايدة من البيانات و بغيّة الحصول على أداء أفضل من ناحية زمن التّنفيذ وسنستعرض فيما يلي بعضاً من أهمّها.

ناقشت الدّراسة [3] مشكلة الحجوم الكبيرة والمتزايدة لصور الاقمار الصّناعيّة الكبيرة وصعوبة الوصول اليها ومعالجتها على مخدّمات مركزيّة، واقترحت استخدام النّموذج البرمجي MapReduce[4] لأجل هذه الغرض مع أخذ تطبيق اكتشاف الحواف كمثال عمليّ، حيث تمّ تقسيم الصّورة الى مُقتطعات وأجزاء ويُعطى كل جزء الى جهاز مُعيّن لتتم معالجته وفق التّطبيق المطلوب تنفيذه. ولأجل ذلك يُقسَّم التّطبيق الى مهمّات تُسند الى العقد العاملة بحيث تُنجز كل مهمّة جزءاً من العمل الكلّيّ بشكل تفرّعيّ على التّوازي، بعد ذلك تُجمع النّتائج مع بعضها البعض وتدمج لتشكيل الخرج النّهائيّ. هذا وقد تناولت الدّراسة آليّة تقسيم الأدوار بين كل من عمليّتيّ Map و Reduce وتنسيق تبادل البيانات فيما بينهما للحصول على النّتائج المرغوبة.

أظهرت النّتائج تحسيناً كبيراً في زمن التّنفيذ للنظام المُقترح بالمقارنة مع منهجيّة الجهاز الواحد وكفائة منهجيّة النظام التّفرّعي في انجاز العمل المطلوب على التّوازي.

ولا تقتصر امكانيّة استخدام هادووب بنمّوذجه البرمجيّ التّفرعيّ على التطبيقات البسيطة، بل تتجاوز ذلك لتنفيذ تطبيقات تعلّم الآلة، وقد بيّن البحث [5] كيف يمكن استثمار هادووب في معالجة بيانات الاستشعار عن بعد لاستخراج السّمات وتصنيف الصّور واستخدموه في معالجة صور الأقمار الصّناعيّة الكبيرة لتحسين اداء وكفائة العمل. قام الباحثون بتحقيق خوارزميّة شجرة القرار لتصنيف الصّور اعتمادا على قيم المتوسّط والتّباين والمسافة الإقليديّة للبكسلات بشكل تفرّعيّ باستخدام MapReduce، حيث تمّ تقطيع الصّورة الواحدة الى كُتل متساوية الحجم ومعالجة كل منها بشكل منفرد ليتمّ في النّهاية تجميع الخرج وتكوين الصّورة النّهائيّة، وقد بيّنت النّتائج أداءً مُنخفضاً للنظام عند معالجة عدد قليل من الصّور وجيّداً عندما كان عددها كبيراً.

تناولت دراسات أخرى تطبيقات معالجة صور ذات طابع مُختلف، فقد تناول الباحثون في [6] تحقيقاً لخوارزميّة العنقدة الشّهيرة K-Means في تصنيف صور الأقمار الصّناعية باعتبارها خوارزميةً تتطلّب قدرات حوسبة عالية وخصوصا عندما يكون حجم الصور كبيراً أو أنّ التّطبيق يحتاج لنتائج خلال وقت محدود جدّاً، ولكون هذه الخوارزميّة ذات طبيعة تكراريّة فإنّ اتّباع المنهجيّة المُستخدمة في [3] و [5] اعتماداً على النّموذج البرمجيّ MapReduce لن يكون مناسباً أبداً بسبب الزّمن الكبير المُستهلك في عمليّات القراءة/الكتابة من/الى نظام الملفّات المُوزّع أثناء تسليك البيانات بين عمليّتيّ Map و Reduce، يُضاف الى ذلك الوقت الاضافي النّاتج من عبئ عمليّات النّسخ المتماثل على كتل البيانات في نظام الملفّات المُوزّع أثناء الكتابة على وسائط التّخزين. بسبب ما سبق فقد اختِيرَ Spark بدلاً عن Hadoop لتجنّب عمليّات القراءة والكتابة التّكراريّة ونُفّذ التّطبيق على عنقود في بيئة حوسبة سحابيّة واختُبر أداؤه بإجراء التجربة عدّة مرّات مع تغيير عدد تكرارات خوارزمية الK-Means وقُيّم أداء النّظام المُقترح مع ال K-Means العاديّة من ناحية زمن التّنفيذ وبيّنت النّتائج أداء عالٍ بشكل كبير لل K-Means التّفرعيّة بالمقارنة مع ال K-Means العاديّة لأجل تكرارات كبيرة.

ليس بعيداً عن الخوارزميّات التّكراريّة و عند معاينة تطبيقات تصنيف أخرى لا بدّ أن نصادف كل من خوارزميّة آلة أشعّة الدّعم SVM[7] و الشبكات العصبونية MLP[8] اللّتين تتميّزان بدقّة أكبر في تصنيف الصّور، انّ كلتا الخوارزميتين مستنزفتين للموارد بشكل كبير خصوصاً اذا كانت البيانات كبيرة الحجم ومتعدّدة الخصائص والسّمات، وهذا ما تطرق اليه البحث [9] حيث استُخدم نظام هادووب للملفّات المُوزّعة لتخزين بيانات الصّور وسبارك كمحرّك تنفيذ لمعالجة صور أقمار صناعيّة عالية الدّقّة باستخدام كل من SVM و MLP. وقد اعتمد منهج البحث على القيام بمعالجة أوّليّة للصّور باعتبار كل بكسل من بكسلات الصّورة المُستخدمة عيّنة تدريب مفردة ثمّ دُرّب النّموذج باستخدام الخوارزميّتين السّابقتين وأُجري تقييمه على بيانات اختبار. بيّنت النّتائج تحسّن في زمن التّنفيذ عند استخدام النّظام المُقترح.

بدراسة الأبحاث السّابقة [3]، [5]، [6] و [9] وجدنا أنّه تم تصنيف الصّور اعتمادا على قيمة البكسل بشكل منفرد، ومعنى ذلك أنّه تم تصنيف البكسل بشكل مستقلّ عن بقية البكسلات المجاورة اعتماداً على قيمه الطّيفيّة فقط دون الاستفادة من قيم البكسلات المجاورة له، وهذا الأسلوب يعطي أداءً جيّداً عند استخدامه على صور منخفضة أو متوسّطة الدّقّة لكنّه ليس مناسباً للصّور ذات الدّقة العالية. إضافة لذلك فإنّ تقنيات التّعلّم العميق هي أكثر تقدّماً من الطّرق السّابقة (بما فيها خوارزميّات تعلّم الآلة) لكونها تستخدم الشّبكات العصبونيّة الالتفافيّة CNN ذات الدّقّة العالية في مجال الرؤية عبر الحاسب والقادرة على استنباط سمات معقّدة للغاية تسهم الى حد كبير في نتائج أفضل لتطبيقات تصنيف الصّور.

من خلال المسح الذي قمنا بإجرائه مؤخّراً على العديد من الدّراسات في مجال معالجة صور الأقمار الصّناعيّة على النّظم فائقة الأداء وتقنيات البيانات الكبيرة بما فيها الدّراسات المذكوروة آنفاً لم نجد أبحاثاً تناولت بشكل واف تطبيق تقنيات التّعلّم العميق على هذه الصّور باستخدام هذه المنصّات، وخصوصاً تلك التّي تقدّم دقّات كبيرة للصور عالية الوضوح، لذلك سعينا في هذا البحث الى المساهمة في هذا المجال من خلال دراسة تحقيق تقنيات التّعلّم العميق والشّبكات العصبونيّة الالتفافيّة اعتماداً على مبدأ الكتلة أو

دفعة بكسلات معاً في معالجة صور الأقمار الصّناعيّة على هادووب و سبارك، كما ناقشنا النّظام المُقترح وآليّة تقييم العمل المُنجز مع النّتائج التّي تم الحصول عليها.

4-2- المعالجة التّفرّعيّة لتطبيق تصنيف الصّور

إنّ كل من عملية تدريب النّموذج أو التّصنيف تتطلب قدرات معالجة عالية خصوصا مع بيانات ذات حجوم كبيرة، الا أنّ الحاجة الى الموارد العالية في تدريب النّموذج هي أكبر بكثير من تلك المطلوبة عند القيام بعمليّات التّنبؤ، كما أنّ تحويل الرمّاز المصدري في عمليّة التدريب من الشّكل التسلسلي الى الشّكل التّفرّعيّ ليس بالأمر السّهل قياسا بعمليّة التّنبؤ على بيانات جديدة لذلك تمّت دراستها بشكل أوسع في هذا البحث.

4-2-1- التّدريب المُوزّع

يتمّ تدريب الشّبكة العصبونيّة تفرّعيّاً وفق نفس المبدأ المُتّبع تسلسيّاً عبر خوارزميّة الهبوط المتدرّج العشوائي(SGD) Stochastic Gradient Descent وهي الخوارزميّة الاكثر شيوعا أثناء التّدريب في مجال الذّكاء الصّناعي ومُشتقّة اساسا من خوارزمية الهبوط المتدرّج Gradient Descent .

ترتكز ال(Gradient Descent) في عملها على ايجاد قيمة المتحول او المتحولات (w) التى تجعل خرج تابع الخطأ المبُيّن في المعادلة (1) أقل ما يمكن. ويمثل تابع الخطأ الفرق بين قيمة الاجابة الفعلية لعينة دخل ما (Y) وقيمة الاجابة المُتنبّئ بها(Y^)

$$Q(w) = \frac{1}{n}\sum_{i=1}^{n} Q_i(w),$$

المعادلة (1) الشكل العام لتابع الخطأ

يشير المتحول (i) الى دليل عينة التدريب الواحدة وبالتالي فان المعادلة السابقة تشير الى مجموع الاخطاء بين القيم المُتنبئ بها والقيم الحقيقية لبيانات التدريب، ويمكن تلخيص آلية عمل الخوارزمية بالخطوات التالية:

1- إعطاء قيم أولية للمتحولات (w)

2- حساب قيمة التنبؤ لعينات التدريب بتعويض قيم الدخل في تابع التّنبؤ (Y^)

3- حساب تابع الخطأ للعينات من خلال المعادلة (1)

4- حساب قيمة مشتق تابع الخطأ $\nabla Q_i(w)$ بالنسبة للمعاملات

5- تحديث قيم معاملات (w) وفق المعادلة التالية(2):

$$w := w - \eta \nabla Q(w) = w - \frac{\eta}{n} \sum_{i=1}^{n} \nabla Q_i(w),$$

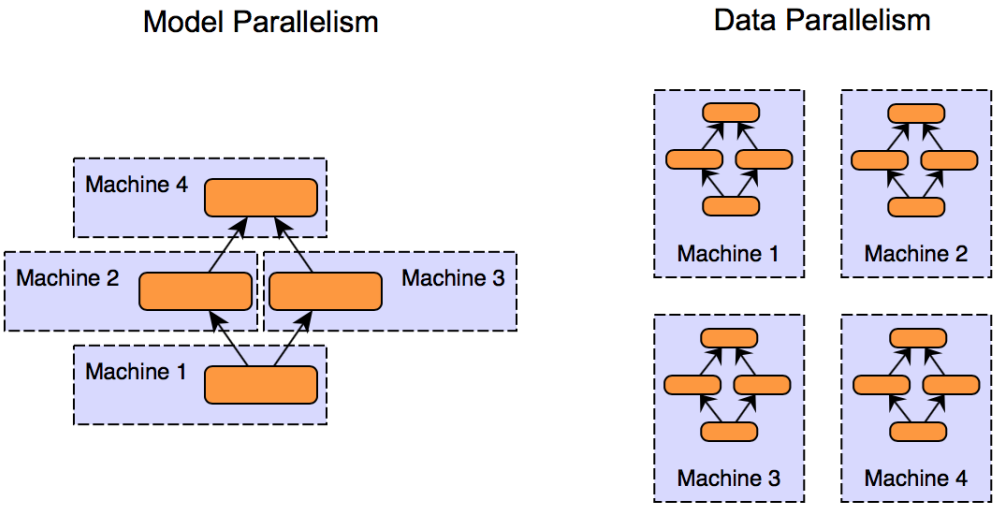
المعادلة (2) تحديث المعاملات

6- تكرار الخطوات من 2 الى 4 عددا من المرّات حتى الوصول الى التقارب (أقل قيمة ممكنة لتابع الخطأ)

تستخدم خوارزمية الهبوط المتدرّج العشوائي(SGD) نفس الآلية السابقة في ايجاد المعاملات المطلوبة (w) وبنفس الخطوات مع فرق أنه يتم أخذ عدد معيّن من عيّنات البيانات تُسمّى دُفعة في كل تكرار عوضاً عن أخذ كل العيّنات بحيث تُؤخذ بشكل عشوائي(Stochastic) وتُطبق الخطوات السابقة آنفاً ثم تتم متابعة العمل على الدفعة التالية.

ان الهدف النّهائي لهذه الخوارزمية هو ايجاد افضل قيم ممكنة للمعاملات التي تمثل أوزان الشبكة العصبونية، وكما رأينا فانّها وفق المبدأ التسلسليّ تتضمّن عدّة جولات من العمل حيث أنّ نتائج كل جولة من التّدريب تُضمّن في النّموذج المُدرّب لتُستخدم في الجولة التّالية، و تُعالج بيانات كل جولة تسلسلياً وفق منهجية الجهاز المنفرد.

أما بالنسبة للتدريب المُوزّع فإنه توجد منهجيّتين معروفتين لتدريب النّموذج بشكل تفرّعي على مجموعة من الأجهزة وهي كما موضّحة في الشكل(1) .



الشكل(1) منهجيّة تفرّعيّة النّموذج مقابل تفرّعيّة البيانات في التّدريب المُوزّع [10]

4-2-1-1- منهجيّة تفرّعيّة النّموذج

بعض نماذج الشّبكات العصبونية كبيرة لدرجة أنّه لا يمكن لذاكرة جهاز واحد (او ذاكرة معالج الرّسوميّات GPU Graphical Processing Unit) أن يستوعب حجمها، لذلك استخدام هذه المنهجيّة هنا هو الحلّ المناسب ومن الأمثلة على ذلك الشبكة العصبونيّة لنظام الترجمة الخاص بغوغل [10].

إنّ تدريب نماذج من هكذا أنواع يتطلّب تقسيم النّموذج الى عدة أقسام على عدّة أجهزة بحيث يتمّ التّدريب على التّوازي وكل جزء يتم تدريبه على عقدة واحدة باستخدام مجموعة البيانات بكاملها، معنى ذلك أنّ طبقات مختلفة من الشّبكة العصبونيّة يتمّ تدريبها على عدّة أجهزة او (GPUs) في نفس الوقت. عند انتهاء جميع العقد من تدريب الأجزاء المخصصة لها يجري تجميع النتائج وفق عملية معيّنة لنحصل في النهاية على النّموذج المُدرّب النّهائيّ. تُسمّى هذه الطّريقة أحيانا في إطار عمل TensorFlow بالنّسخ

المتماثل داخل البيان (in-graph replication) وتُعتبر من الأساليب الصّعبة التّطبيق للحصول على أداء جيّد .

4-2-1-2- منهجيّة تفرّعيّة البيانات

تُعرف هذه المنهجيّة في إطار عمل TensorFlow بالنّسح المتماثل بين البيان (between-graph replication) وفيه يُستخدم النّموذج على كل عقدة لتدريب بيانات مختلفة بخلاف الطريقة السابقة التي فيها يُدرّب جزء من النّموذج على كل البيانات.

تحسب كل عقدة المشتقّات "gradients" الناتجة من حساب الأخطاء في عملية الهبوط المتدرّج (Gradient Descent) لجزء من دفعة بيانات واحدة في كل مرّة، ثمّ يتم تجميعها مع نتائج العقد الباقية في كل تكرار من الخوارزمية كما لون أن العمل يجري على عقدة واحدة حيث أنّه يتعيّن على كل عقدة ان ترسل التغيّرات الى جميع العقد الأخرى. يتم بعد ذلك تحديث المعاملات (اوزان الشبكة العصبونية) عند كل عقدة عاملة وتُكرّر العملية على الدُفعات التالية من البيانات حتى الانتهاء منها.

4-2-1-3- معماريّات التّخاطب والاتّصال في عمليّة التّدريب

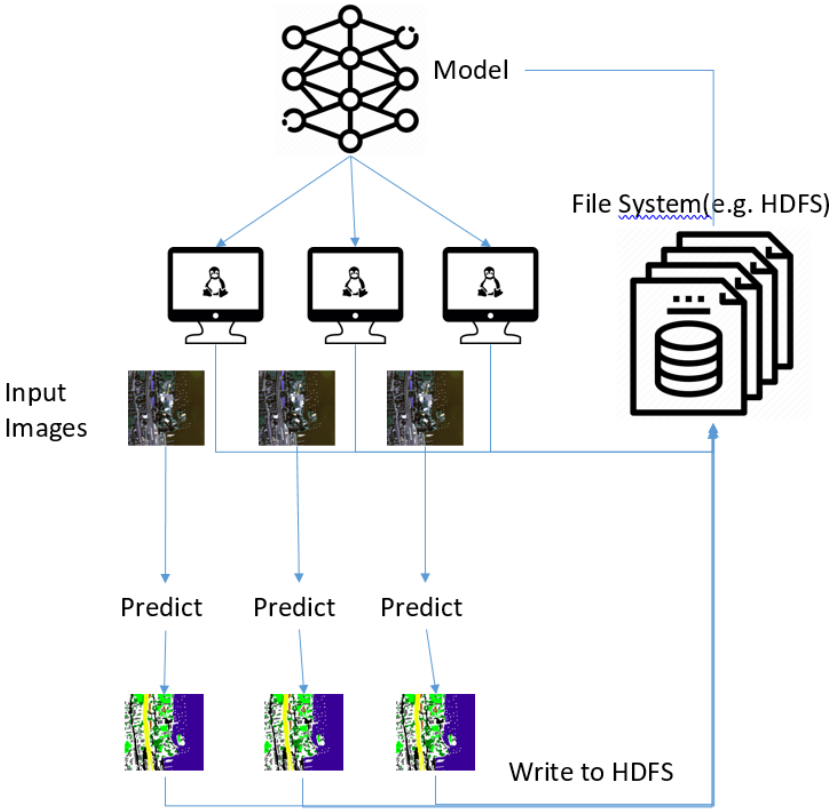
توجد معماريتان لتدريب نمّوذج التّعلّم العميق على التّفرّع الأولى تُدعى **مخدّم المعاملات المركزية** حيث توجد عقدة مركزيّة تعمل كمخدّم يتولّى تحديث معاملات النّموذج واجراء بثّ عام لها الى كافة العقد العاملة. في النّمط المتزامن منها ينتظر المخدّم انتهاء العقد العاملة من حساب المشتقّات وارسالها اليه ليتولّى بدوره حساب قيم متوسطاتها وتعديل أوزان الشبكة العصبونية ثم يرسلها من جديد الى جميع العقد الأخرى مع دفعة جديدة من البيانات لبدء تكرار جديد من خوارزمية ال(SGD)، أمّا في النّمط غير المتزامن فلا تنتظر الأجهزة تحديثات النّموذج من المخدّم في كل دفعة بيانات، بل حيث تعمل بشكل مستقل عن بعضها وتتشارك النّتائج فيما بينها.

المعمارية الثّانية هي الحلقة (Ring) التي تكون فيزيائية او منطقية ضمن عنقود العقد العاملة وفيها لا يوجد مخدّم مركزي لتحديث اوزان الشّبكة العصبونية وانّما تقوم كل عقدة بحساب مشتقات الجزء الخاص بها من دفعة البيانات وارسالها الى العقدة التّالية في

الحلقة واستقبال المشتقّات من العقدة السّابقة في الحلقة، ويجري بعد ذلك تحديث معاملات النموذج على كل عقدة. وبالتالي من أجل N عقدة عاملة ضمن الحلقة، ستكون جميع العقد قد استلمت ال gradients من جميع العقد الأخرى بعد N-1 عملية نقل بحيث تشمل العمليّة الواحدة ارسال واستقبال البيانات.
تُعتبر هذه المعماريّة مثاليّة من ناحية استهلاك عرض الحزمة لكونها تضمن استخدام كامل لعرض حزمة الشبكة خلال رفع وتحميل البيانات على كل عقدة.

4-2-2- التّنبّؤ المُوزّع

لكي نتمكّن من إجراء التّنبّؤ المُوزّع بشكل تفرّعيّ فينبغي أن يتم تحميل النّموذج المُدرّب مُسبقاً الى ذاكرة كل عقدة في العنقود سواء من نظام الملفّات المحلّي للعقدة أو من نظام الملفّات المُوزّع كما في الشكل(2) وفيه تقرأ كل عقدة جزءاً من البيانات فقط تعالجها بشكل مستقلّ بحيث تدخل كل صورة الى النّموذج لتُعالج وينتج من ذلك خريطة التّقطيع الموافقة لصورة الدّخل لتتم كتابتها الى نظام الملفّات المُوزّع.



الشكل(2) آليّة التّنبّؤ الموزّع أثناء تصنيف الصّور

4-3- بنية نمّوذج U-Net

تتألّف معمارية شبكة ال U-Net كما يبيّن الشكل(3) من جانبين، أيسر يُسمّى مسار التّقليص وأيمن يسمّى مسار التّوسيع. يتبع مسار التقليص المعمارية النموذجيّة لشبكات الطيّ او الالتفاف (convolution) ويتألّف من التّطبيق المتكرّر للالتفاف بمصفوفات ذات أبعاد 3*3 من دون حشو، ويعني ذلك انّ أبعاد مصفوفات الخرج مختلفة عن الدّخل، متبوعاً بتابع تفعيل من النّوع Rectified Linear Unit(RLU) أو تابع التصحيح الخطّي الذي يعطي خرجاً مساوياً للدخل اذا كان موجباً، و صفرا اذا كان سالباً. كما يتضمّن النموذج عمليّة تجميع أعظمي بمصفوفة من قياس 2*2 وخطوة 2 لأجل عمليّة الاختزال [11].

في كل خطوت الاختزال (Down sampling) نقوم باختزال عرض وارتفاع المصفوفات وبمضاعفة عدد القنوات.

كل خطوة من مسار التّوسيع تتالّف من:

1- تضخيم خرائط السّمات (feature maps) متبوعة ب بطبقة التفاف (convolution) 2*2 تخفّض عدد قنوات السّمات الى النّصف
2- دمج الخرج الناتج من الخطوة 1 بخريطة السّمات المقابلة لها من مسار التّقليص وذلك بعد تعديل مقاسات ابعادها من خلال القطع لتوافق عملية الدّمج
3- طبقتي التفاف 3*3 متتاليتين متبوعا بتابع التنشيط (RELU) لكل منهما

في آخر طبقة توجد عملية التفاف 1*1 تُستخدم لمقابلة خرائط السّمات مع عدد الاصناف المُختارة سابقا ولكي تكون خرائط السّمات في الخرج النهائي للنموذج المُدرّب ملتحمة من دون تشقّقات في الالتحامات فإنّه من المهم اختيار ابعاد قياسات صور الدخل بحيث تُطبّق عمليّات التجميع الاعظمي 2*2 على مصفوفات بابعاد زوجية [11].

نظراً لأنّ عملية الالتفاف(Convolution) المُستخدمة في استخراج السّمات أثناء معالجة البيانات في الشبكة العصبونيّة لهذا النّموذج هي من النوع الذي لا يحوي حشو فإنّ صورة الخرج تكون ذات أبعاد أقل من صورة الدّخل بمقدار ثابت بالحالة المعيارية لتطبيق النّموذج، الّا أنّه في حالتنا هذه استخدمنا نمّوذج مُعدّل من U-Net بحيث استُخدمت طبقة التفاف مع حشو للمحافظة على أبعاد صورة الخرج، مما يعني ان صورة الدّخل وصورة الخرج النهائية لهما نفس الأبعاد.

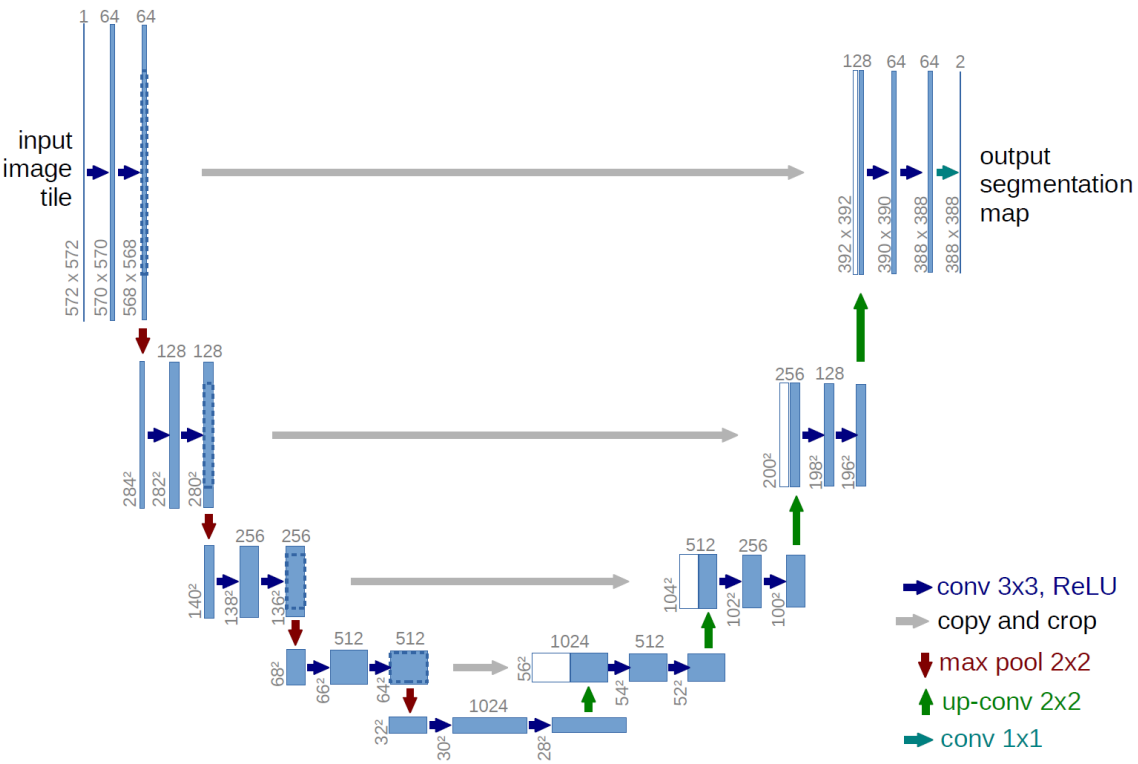
بالمجمل احتوت معمارية نموذج الU-Net المُستخدم في هذه البحث على **23 طبقة** من طبقات الالتفاف، وتتضمّن هذه الطّبقات ما يبلغ **31،053،225** معامل (بارامتر) . وهو من النّماذج الكبيرة نسبيّاً التّي تتطلب وقتا طويلا للقيام بعمليّة التّدريب خصوصا اذا كانت بيانات التدريب كبيرة الحجم.

تمّ اقتراح هذا النمّوذج للعمل عليه لعدّة اسباب وهي:

1- أحد نماذج تقنيات التّعلّم العميق ذات الأداء الجيّد للقيام بعملية تصنيف الصّور وفق الخرج المطلوب بدقّة تصل الى 95.40 لبيانات التّدريب و 83.354 لبيانات الاختبار

2- إمكانية التّطبيق على صوركبيرة الحجم من خلال تقطيع الصورة الواحدة الى أجزاء بحيث كل جزء يدخل الى النّموذج ليتم تدريبه بشكل منفصل وهذه الطّريقة مناسبة جدّا اذا كانت الصورة كبيرة جدّا (من فئة الغيغا بايت)

3- لا توجد حاجة لمجموعة كبيرة من الصّور للقيام بالتدرّب عليها لانّه يمكن استخدام صورة واحدة وتقطيعها الى قطع متداخلة مع بعضها البعض لتوليد مجموعة كبيرة من أمثلة التدريب التّي تدخل نموذج التّدريب



الشكل(3) معمارية ال U-Net[11]

4-4- استخدام سبارك بدلا عن هادووب كمحرّك أساسيٍّ للتّنفيذ

على الرّغم من أنّ هادووب هو المنصّة الأشهر في مجال معالجة البيانات الكبيرة بسبب تقديمه كل من نظام الملفّات المُوزّع ومعمارية MapReduce للمعالجة التفرّعيّة المُخصّصين أساساً للعمل عليه واللّذين يعملان بكفائة مع العديد من المهام الّا أنّه يعاني من مشكلة الأداء البطيئ والسيّى أحيانا من أجل التطبيقات التكراريّة أو التفاعليّة، وهي تطبيقات تحتاج للوصول الى البيانات بشكل متكرّر أثناء عمليّات المعالجة سواء كانت البيانات نفسها التي تمّت قرائتها أول مرّة او بيانات نتجت من عمليّات معالجة في خطوة سابقة مثلاً.

تُعرف هذه المشكلة أحياناً بمشكلة البطئ في تشاركيّة البيانات في نمّوذج MapReduce على هادووب بسبب الوقت المُستهلك في عمليّات القراءة والكتابة من/الى نظام الملفّات بالإضافة الى عمليّات تسليك البيانات (Serialization) و التّضاعف المتماثل في نظام الملفّات المُوزّع، ووفقا لقيم الأداء للعديد من تطبيقات MapReduce فإنّ 90% من وقتها مُستهلك في عمليّات القراءة و الكتابة من/الى HDFS [12] [13].

وعلى النقيض من ذلك يستخدم سبارك ما يُعرف بمجموعات البيانات المرنة (Resilient Distributed Dataset) التي المعروفة اختصاراً ب RDD التّي تقدّم خدمة وضع البيانات في الذاكرة مباشرة لتتم معالجتها وتُسمّى احياناً بالحساب – في- الذّاكرة. يعني ذلك أنّه يتم تخزين حالة الذّاكرة ككائن (object) تتم مشاركته بين الأعمال أو التّطبيقات الأخرى العاملة. هذه التّقنية هي أسرع ب 10 الى 100 مرّة من تلك التّي تعمل اعتمادا كلّيّاً على التّناقل عبر الشّبكة ووسائط التّخزين كما في هادووب لذلك اعتُمد استخدام سبارك في عمليّات تنفيذ المعالجة الفعليّة للبيانات فيما بقي استخدام نظام الملفّات الموزّع الخاص بهادووب للتخزين.

4-5- التّدريب المُوزّع على سبارك

تدعم واجهات برمجة التطبيقات (API: Application Programming Interfaces) الخاصّة بسبارك العديد من خوارزميّات تعلّم الآلة بما في ذلك بناء وتدريب شبكات عصبونيّة بالإضافة الى API أخرى لعمليّة التنبّؤ، الّا أنّه يفتقر الى الأدوات والمكتبات المطلوبة التّي تمكّنه من التّعامل مع تقنيات التعلّم العميق لتلبية احتياجات التّطبيقات، من هنا قامت عدّة شركات بالاضافة الى مجتمع البيانات الكبيرة بملئ هذا الفراغ وتقديم عدّة مكتبات لحل هذه المكشلة.

من بين عدّة أدوات وبرمجيّات تدمج تقنيات التّعلّم العميق مع سبارك فأنّ عدداً قليلا منها فقط يدعم تدريب الشّبكات العصبونيّة العميقة على التّفرّع، هنا نجد مكتبات TensorFlowOnSpark[14] ، Horovod[15] ، BigDL[16] ، و Deeplearning4j[17] أمثلة عن هكذا أدوات تتيح استخدام تقنيات التّعلّم العميق على الانظمة التفرّعيّة بشكل سهل وعمليّ.
لأجل بحثنا هذا اخترنا إطاري عمل TensorFlowOnSpark و Horovod لأن كليهما يستطيعان التّعامل بشكل فعّال مع مكتبة TensorFlow الشّهيرة لبناء الشّبكات العصبونيّة وتدريبها باستخدام لغة بايثون مع امكانية استثمار العديد من المكتبات المساعدة الاخرى في العمليّات على المصفوفات، ايضا لا توجد تعقيدات كبيرة في اعداد بيئة العمل وتهيئتها.

4-5-1- إطار العمل تنسورفلو على سبارك (TensorFlowOnSpark)

إطار عمل مُطوّر أساساً من قبل شركة ياهوو للقيام بتدريب مُوزّع لنماذج التّعلم العميق بنطاق واسع على عناقيد هادووب ضمن البيئة السّحابية الخاصة بالشركة نفسها وتمكننا من تنفيذ تطبيقات TensorFlow على عناقيد Spark و Hadoop. ويعتمد إطار العمل هذا على خدمة غوغل للاتّصال عن بعد (gRPC: Google Remote Procedure Call) كتقنية لتبادل البيانات أثناء التّدريب وهي مشتقة من البروتوكل الاساسي في الاتّصال عن بعد (RPC: Remote Procedure Call) الذي يتكوّن من برنامجين هما المخدّم والزّبون بيحيث يعرض المخدّم خدماته على برنامج الزبون بشكل توابع برمجية

تُستدعى عن بعد [18]. وهذه التقنية هي الاساس الذي تعتمد عليه العقد العاملة ضمن العنقود في الاتصال وتبادل البانات فيما بينها.

4-5-2- إطار العمل هوروفود (Horovod)

إطار عمل مفتوح المصدر مُطوّر من شركة **Uber** و يُستخدم للتدريب المُوزّع لنماذج التّعلّم العميق ويعمل مع عدة تقنيات وأطر أخرى هي **TensorFlow** و **Keras** و **PyTorch** و **Apache MXNet**. اعتُمد في شركة Uber هذا الإطار على عدة تقنيات أثناء تطويره حيث اختير نمّوذج واجهة تمرير الرسائل المعروفة اختصارا باسم MPI لتبادل البيانات بين المعالجات الذي يُعتبر اسلوب مباشر وسهل التعامل معه [15].

4-6- تدريب النّموذج

في هذه العمليّة يكون لدينا صور الدّخل مع خرائط التّقطيع المُقابلة لها لتدريب الشبكة العصبونية بتطبيق الهبوط المتدرّج العشوائي.

يوجد معامل مهمّ ينبغي تحديده قبل البدء بعمليّة التّدريب وهو قياس دفعة البيانات الكلّيّ (Global Batch Size) سنرمز لها بGBS، ويُقصد به حجم الدّفعة التي ستتشارك العقد العاملة في معالجتها في وقت واحد ويمثّل مجموعة الدّفعات الجزئيّة التي تقوم كل عقدة بمعالجة واحدة منها فقط. يمكن أن نبقي قيمة الGBS عند التّدريب على التّفرّع مماثلا لقيمة حجم الدّفعة عند التّدريب على عقدة واحدة وهنا ستتقاسم العقد دفعة البيانات لتعالج كل منها جزء فقط، ويمكن أيضاً جعل قيمة الGBS تساوي قيمة حجم الدّفعة مضروبة بعدد العقد أي $GBS = N * (batch\_size)$، وهذه الطّريقة فعّالة للغاية لناحية أنّه يمكن استخدام حجم دفعة كلّيّة كبير جدّا ولا تستطيع معه العقدة الواحدة استيعابها عند المعالجة نظراً للحاجة الى ذواكر كبيرة جدّاً، ومن المعلوم ايضاً أنّ الدّقّة بهذه الحالة تصبح أفضل عادة مقارنة بالحالة الأولى لأنّ تحديث أوزان الشّبكة العصبونيّة سيجري بعد تدريبها على فضاء عيّنات أكبر.

إنّ هذه الميّزة تجعل من النّظم الموّزّعة تملك **قيمة مُضافة** الى جانب السّرعة في الأداء مقارنة بمنهج الجهاز الواحد نظراً لامكانيّة تدريب النّموذج باستخدام معاملات لم يمكن بالامكان استخدامها لو تمّ التّدريب بدونها.

4-7- العتاد والبيئة المُستخدمة

لبناء العنقود تم تجهيز مخدّم بالقدرات والموارد المُتاحة لانشاء عنقود من الاجهزة المتصلة مع بعضها البعض حيث تمّ انشاء 4 أجهزة افتراضية على مخدّم يعمل بذاكرة GB 64 ومعالج متعدّد النّوى يعمل بتردد GH 2.1 مُنصّب عليه بيئة افتراضية Xen Server 7.3 ،وتمّ تخصيص لكل آلة افتراضيّة معالج ثماني النوى وهذه الآت متصلة مع بعضها البعض بشبكة محلية افتراضية. المهم ذكره هنا أنّه تم تنصيب اتصال التشفير الآمن Secure-Shell(SSH) على جميع العقد العاملة لكي تتمكّن من الاتّصال وتبادل البيانات فيما بينها بشكل آمن لتجنّب القيام بعمليّة تسجيل الدّخول في كل عمليّة اتّصال، مع القيام بنسخ مفاتيح التشفير الى العقدة الرّئيسيّة، وهو أحد الاعدادات الاساسيّة اللّازمة لعمل كل من Spark و Hadoop.

لا يحوي المخدّم أي معالجات رسوميات Graphical Processing Unit(GPU) لذلك تم الاعتماد كلّيّاً على وحدة المعالجة المركزية للمخدّم، كما تمّ أيضاً تفعيل ميّزة تثبيت وحدة المعالجة المركزيّة (Virtual CPU Pinning) في البيئة الافتراضية للمخدّم وهي ميّزة تمكّن من مقابلة النّوى الفيزيائيّة للمخدّم مع نوى معالجات الآلات الافتراضية لتحقيق أكبر قدر ممكن من عزل الموارد بين تلك الآلات، وهو اعداد مهمّ في حالتنا هذه من أجل محاكاة نظم البيانات الكبيرة في بيئة افتراضيّة أُنشئت أساساً لتحقيق أكبر قدر من تشاركيّة الموارد.

بعدها تم تركيب منصة Spark على كل العقد وإعداداها بمتحولات البيئة اللازمة لعمل العنقود كما تمّ انشاء بيئة بايثون افتراضية نسخة 3.7 و تحميل المكتبات المطلوبة من أجل عملية التّنفيذ، ومن بين أهم ما تم تنصيبه مكتبة TensorFlow وإطاري عمل TensorFlowOnSpark و Horovod المستخدمين في عملية التّدريب الموزّع، فيما استُخدم

نظام الملفّات الموزع Hadoop بعد تركيبه على العقد جميعها من أجل تخزين بيانات التدريب وتخزين المودل وصور الخرج النهائية عند القيام بالتنبّؤ.

يوضّح **الجدول(1)** تفاصيل خصائص العقد العاملة في العنقود، وبعد الانتهاء من جميع الخطوات السّابقة تصبح البيئة جاهزة لتجريب البرنامج/البرامج المطلوب.

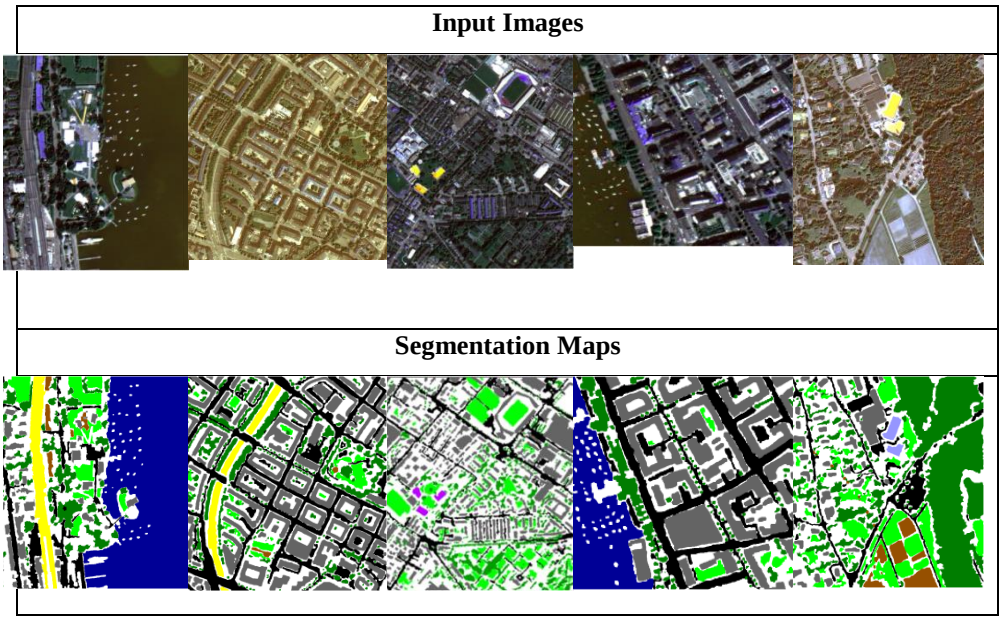
الجدول(1) خصائص العنقود والعقد العاملة ومواصفاتها الرّئيسيّة

عدد العقد	من 1 الى 4
نظام التشغيل في العقدة	Centos 7
ذواكر العقدة	16 غيغا بايت للعقدة الرئيسيّة و14 غيغا بايت لكل عقدة عاملة
عدد نوى المعالج في كل عقدة	8
هادووب	نسخة 3.1.1 (توجد عقدة واحدة لأسماء النطاق NameNode على العقدة المركزيّة أمّا بقية العقد فهي عقد بيانات DataNode)
سبارك	نسخة 3.0.0 ويعمل بالنمط المستقلّ
لغة بايثون	نسخة 3.7.7

تمّ التنفيذ العمليّ لكل من تطبيقي التّدريب والتّنبّؤ باستخدام اطاري العمل TensorFlowOnSpark و Horovod حيث أجريت التّجارب لكل إطار عمل على حدا على Spark و Hadoop وقُيّم أداء كليهما من ناحية زمن التّنفيذ كمعامل أساسي لقياس الأداء.

4-8- مجموعة البيانات

تحوي مجموعة البيانات المُستخدمة على 14 صورة دخل يقابلها 14 صورة من خرائط التّقطيع حيث يوضّح الشكل (4) بعضاً منها. تختلف الصّور الموجودة فيما بينها من ناحية الحجم ويبلغ أكبرها 17 ميغابايت وهي ذات أبعاد متنوّعة. تتألّف كل صورة من المجموعة من 4 حزم طيفيّة هي الأحمر Red والأخضر Green والأزرق Blue اضافة الى طيف الأشعة تحت الحمراء Near Infrared Radio ليصبح التشكيل يضم القنوات الاربعة معاً (**R, G, B, NIR**) علماً ان ال (NIR) لا يمكن رؤيته بالعين البشريّة المجرّدة لكنه يحوي معلومات مهمة يمكن الاستفادة منها، أمّا خرائط التّقطيع فتحوي على 9 قنوات يمثّل كلٌّ منها صنفاً من الأصناف ال9 التي تقابل أنواع الغطاء الأرضي المُستهدفة في تطبيقنا وهذه الأصناف هي الطّرق، الأبنية، الأشجار، العشب، التربة العارية، المياه، السّكك الحديدية، المسابح اضافةً الى الّلاصنف



الشكل (4) مجموعة بيانات التّدريب

4-9- عملية توليد الصّور

نظراً لانّ عدد الصّور المتاحة هي قليلة (14 صورة تدريب) تم اللّجوء الى طريقة تقطيع الصّورة الواحدة الى عدّة مُقتطعات (blocks) متداخلة فيما بينها بابعاد 4*128*128 مع قياس خطوة يبلغ 48 بكسل للانتقال من مُقتطع الى المُقتطع الذي يليه، وهذه الطريقة تمكننا من توليد عدد كبير من الصور الجزئية(مُقتطع) المُتداخلة التى يمكن استخدامها في عملية التّدريب حيث ينتج في حالتنا اكثر من 6700 مقتطع للتدريب في حال استخدام خطوة بحجم 48 بكسل واكثر من 15000 مقتطع في حال استخدام خطوة بحجم 32 بكسل، وهذه العدد الكبير هو من دون استخدام تقنيات تضاعف البيانات(Data Augmentation) الشائع استخدامها في مجال تعلّم الآلة عند الرّغبة بزيادة كمية بيانات التّدريب. بهذه الطريقة اصبح لدينا عدد كاف من بيانات التّدريب الجاهزة لملائمة المودل.

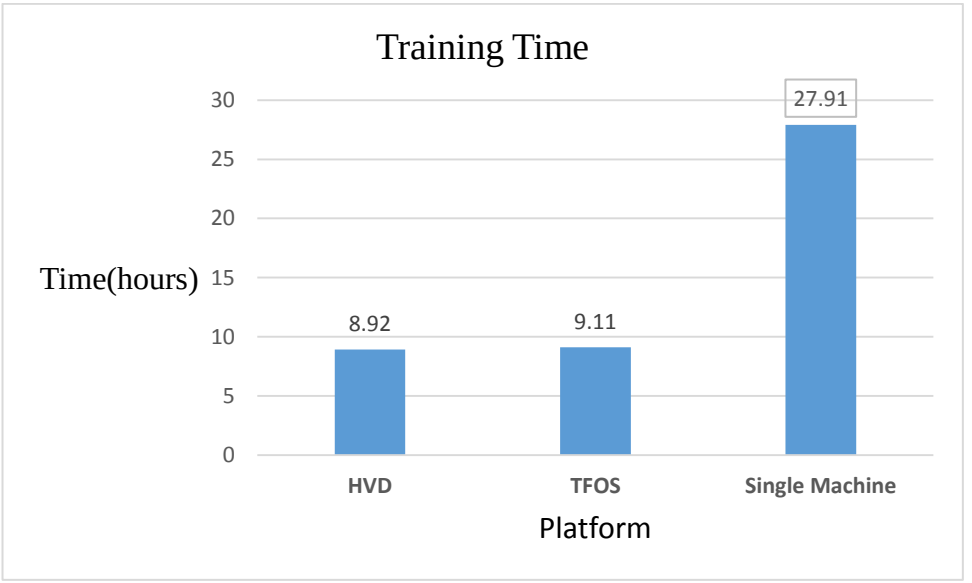
4-10- عمليّة التنبّؤ

في هذه العمليّة يجب أن يكون النّموذج موجودا في نظام الملفّات المحلّي لكل عقدة على حدى أو يكون موجوداً في نظام الملفّات المُوزّع ضمن العنقود او العناقيد ان وجدت وتبدأ العمليّة بتحميل النّموذج الى الذّاكرة الرّئيسيّة ويتم تقسيم البيانات على العقد بحيث تقوم كل واحدة منها بعملية التنبّؤ والحصول على الخرج وكتابته الى HDFS. نظراً لعدم توفّر صور كافية للاختبار في عمليّة التّنبؤ، تمّ مضاعفة إحدى الصّور من خلال نسخها عدّة مرّات لتكوين عدد كبير منها ومعالجتها في التّطبيق، وهذا مقبول في حالتنا لكوننا معنيين بقياس الزّمن المُستهلك لانجاز العمل المطلوب كمعامل أداء أساسي بغضّ النّظر عن محتوى الصّور التي تتم معالجتها. الجدير بالذّكر هنا أنّه لا حاجة لتعديل حجم صورة الدّخل الى 4*128*128 كما كانت صور الدّخل في عمليّة التّدريب، بل يقبل النّموذج صوراً من أبعاد متعدّدة بشكل مُتكيّف مع إبقاء عدد القنوات مساوياً ل4.

5- النتائج والمناقشة

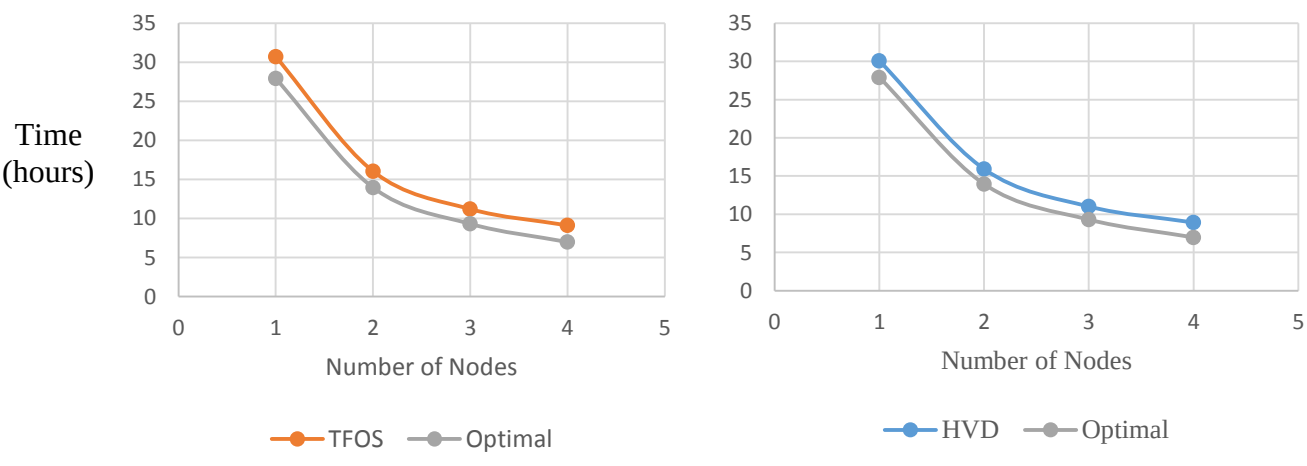
تمّ تنفيذ كل من عمليتي التّدريب والتنبّؤ على العنقود وفق مراحل لمعرفة وتقييم كفائة استخدام إطار العمل Spark في انجاز العمل بالإضافة الى قابلية التّوسّع للبيئة التي تم تهيئتها والعمل عليها حيث تمّ الاختبار على عنقود بحجم 1 و2 و3 و 4 عقد على التوالي وقياس الزمن المستهلك الكلّي الذي استغرقه العنقود لانجاز العمل المطلوب.

بدايةً تمّ تدريب النّموذج على عنقود بحجم 4 باستخدام كل من TensorFlowOnSpark و Horovod وتدريبه أيضاً وفق منهجيّة الجهاز المنفرد وقورنت قيم الأداء لكل منهم. أعطى تدريب النّموذج على منصّات البيانات الكبيرة و النّظم المّوزّعة نتائج جيّدة جدّا لناحية التّحسين الكبير في الزمّن المُستهلك حيث أنهى العنقود تدريب النّموذج خلال 8 ساعات و55 دقيقة باستخدام Horovod فيما استغرق 9 ساعات و7 دقائق باستخدام TensorFlowOnSpark، أمّا منهجيّة الجهاز المنفرد فقد استهلكت 27 ساعة و 55 دقيقة لاستكمال التّدريب كما في الشكل (5).



الشكل (5) زمن تنفيذ التّدريب باستخدام منهجيّة الجهاز المنفرد مقابل منهجيّة النّظم المُوزّعة على منصّات البيانات الكبيرة

يمكن القول أنّ هذه النّتيجة متوقّعة بالفعل لأنّ استثمار قدرات عدّة أجهزة (عقد) تعمل معاً لانجاز مهمّة واحدة مُشتركة ضمن نظام مُوزّع سيفضي الى أداءٍ أفضل بكثير عند المقارنة مع العمل على جهاز منفرد ذو مواصفات مطابقة لعقدة واحدة، كما أنّ الأداء سيتحسّن في كلّ مرّة يزداد حجم العنقود وفق نفس الشّروط الأوليّة للنظام. ولمعرفة كفاءة النّظام المُقترح في إنجاز هكذا أنواع من العمل خصوصاً عندما تكون حجوم البيانات كبيرة جدّا فإنّه ينبغي قياس أداء النّظام بشكل مستمرّ مع زيادة عدد العقد في العنقود لمعرفة اذا ما كان هناك أداء خطّي للنظّام أو ما يقاربه بزيادة عدد العقد من خلال متابعة الفرق بشكل مستمر. وتجري هذه المقارنة عادة مع الحالة المثاليّة للأداء التّي تُؤخذ قيمها اعتماداً على زمن تنفيذ العقدة الواحدة T ومن ثمّ تُحسب أزمنة التنفيذ على عقدتين بتقسيم T على 2 وتُحسب على 3 عقد بتقسيم T على 3 وهكذا دواليك حتى الانتهاء من التّجربة باستخدام حجم العنقود الأعظميّ. هذا وقد تمّ تسجيل النّتائج التّي تمّ الحصول عليها في كلّ مرّة وتمثيلها كما يبيّن الشكل(6).

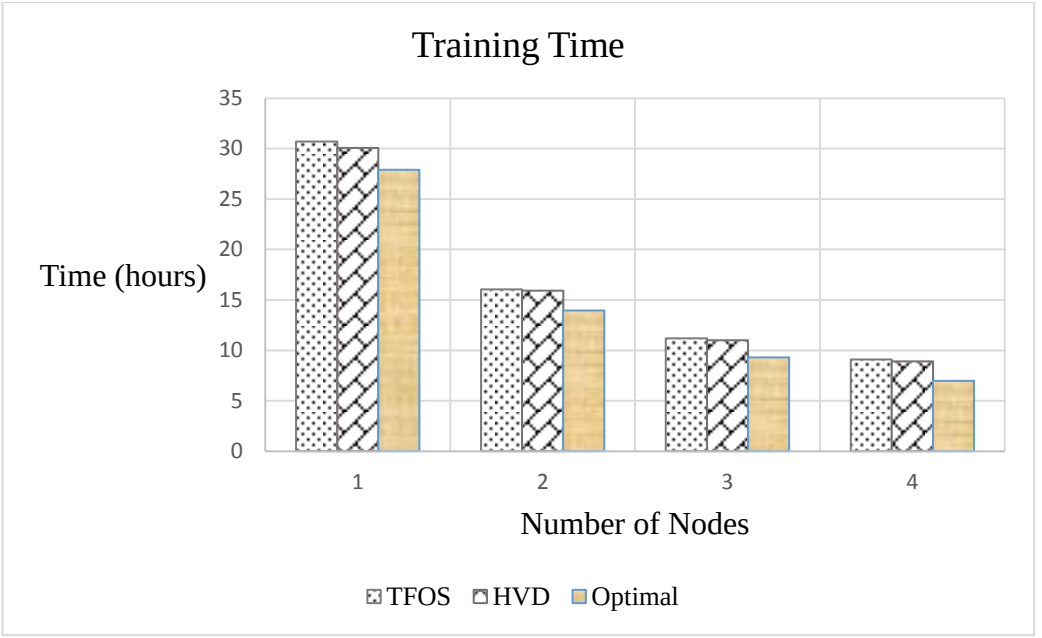


الشكل(6) زمن تدريب نمّوذج ال U-Net على العنقود بالمقارنة مع الحالة المثاليّة بزيادة عدد العقد

يوضّـح الشـكل(6) مقارنــة بـين أداء عمـل TensorFlowOnSpark و Horovod مـع الحالة المثالية من حيث زمن التّنفيذ عند تدريب النّموذج مع زيادة عدد العقد المستخدمة ضمن العنقود بحيث تمّت إعادة التجربة من أجل كل عدد جديد من العقد وحساب الزمّن فـي كلّ مرّة ليتبيّن لنـا أن هنـاك فـرق بأقل مـن 3 سـاعات عنـد وجـود عقدة واحـدة فقـط ليتوسـع هـذا الفـارق قليلاً عنـد وجـود عقدتين فيمـا يسـتقر الوضـع نسـبيا عنـد زيـادة العقـد. إنّ سبب زيادة الزمن يعود الى العبـئ الإضـافي النـاتج من عميات الاتّصـال بين العمليّات وتبـادل البيانـات فيمـا بينهـا بعـد الانتهـاء مـن حسـاب المشـتقّات فـي كـل دفعـة بيانـات (batch) لكي تحصل جميع العمليّات على نفس القيم اللّازمة لتحديث النّموذج.

نستنتج هنا أنّ هناك عبئاً اضافياً من تنفيذ التّدريب على Spark وخصوصا عندما يكون العمل يجري على عدد قليل من العقد العاملة في العنقود الّا أن التحسين في زمن تنفيذ يبدو جليّا عندما يزداد حجم العنقود وهو يعطي أداء جيّدا ومقبولاً لأنّ هامش الفرق بين الحالة المثاليّة بالمقارنة مع إطاري العمل تبقى ثابتة تقريباً حيث نلحظ ذلك من كون أنّ الخطين البيانيّين لكل من TFOS و HVD لا يتباعدان عن الخطّ البياني للحالة المثاليّة مع زيادة عدد العقد.
يعود الفرق بين الخطّ البياني للحالة المثاليّة والخطّين البيانيين الخاصين باطاري العمل الى العبئ الاضافي النّاتج من تهيئة واعداد كل من Horovod و TensorFlowOnSpark إضافة للوقت اللازم في إعداد العنقود بشكل عام، يُضاف اليه الزّمن المُستهلك في الاتّصالات وتبادل البيانات بين العقد العاملة.

يمكن مشاهدة الفرق بين إطاري العمل من خلال خلال إعادة تمثيل الشّكل السّابق باستخدام مُخطّط آخر كما يبيّن الشكل(7) ، حيث نلحظ التّقارب الكبير بين أدائي إطاري العمل مع فارق بسيط لصالح Horovod .

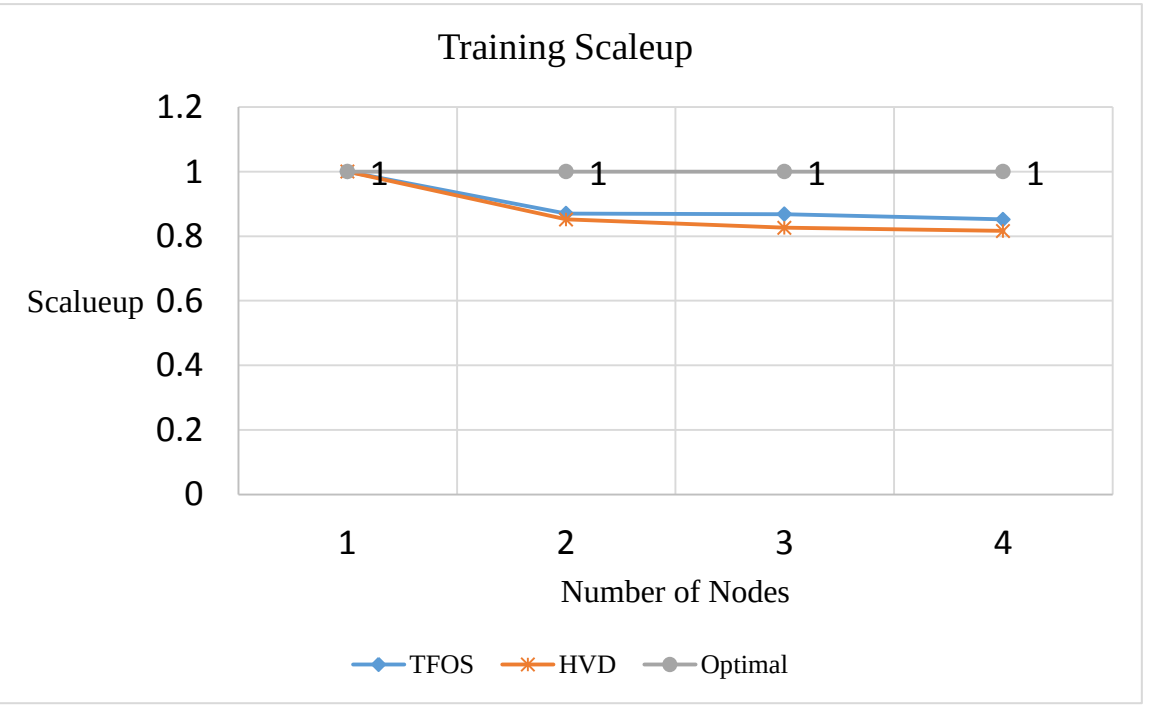


الشكل(7) مقارنة بين إطاري العمل TensorFlowOnSpark و Horovod من ناحية زمن تدريب النّموّذج على العنقود مع الحالة المثاليّة

أحد المؤشّرات المهمّة التّي ينبغي النّظر اليها بما يتعلق بأداء هكذا اختبارات هو معامل التّوسع (Scaleup) ، ويُقصد به قياس الأداء حين يتمّ زيادة الموارد المتاحة للتنفيذ مع زيادة حجوم البيانات المّراد معالجتها بنفس النسبة، وتمّ ذلك في دراستنا من خلال معالجة 25% من البيانات بعقدة واحدة، ثمّ معالجة 50% بعقدتين، ثمّ 75% بثلاث عقد لننتهي بمعالجة كل البيانات بأربع عقد، ونحصل على قيم التّوسّع من خلال تقسيم زمن التّنفيذ بعد زيادة البيانت وحجم العنقود على زمن التّنفيذ قبل الزّيادة.

يبيّن الشكل(8) مقارنة بين الحالة المثالية للتّوسّع والتي يبقى فيها الزمن المُستهلك ثابتاً تماما بتغير الموارد المُتاحة وحجوم البيانات مع حالتنا باستخدام أُطر العمل وفق نفس القيم السّابقة للتجربة، حيث نلحظ انخفاض في معامل التّوسّع لكل من إطاري العمل عند استخدام عقدتين مع استمرار الانخفاض بنسبة بسيطة عند استخدام 3 عقد، فيما يستقرّ

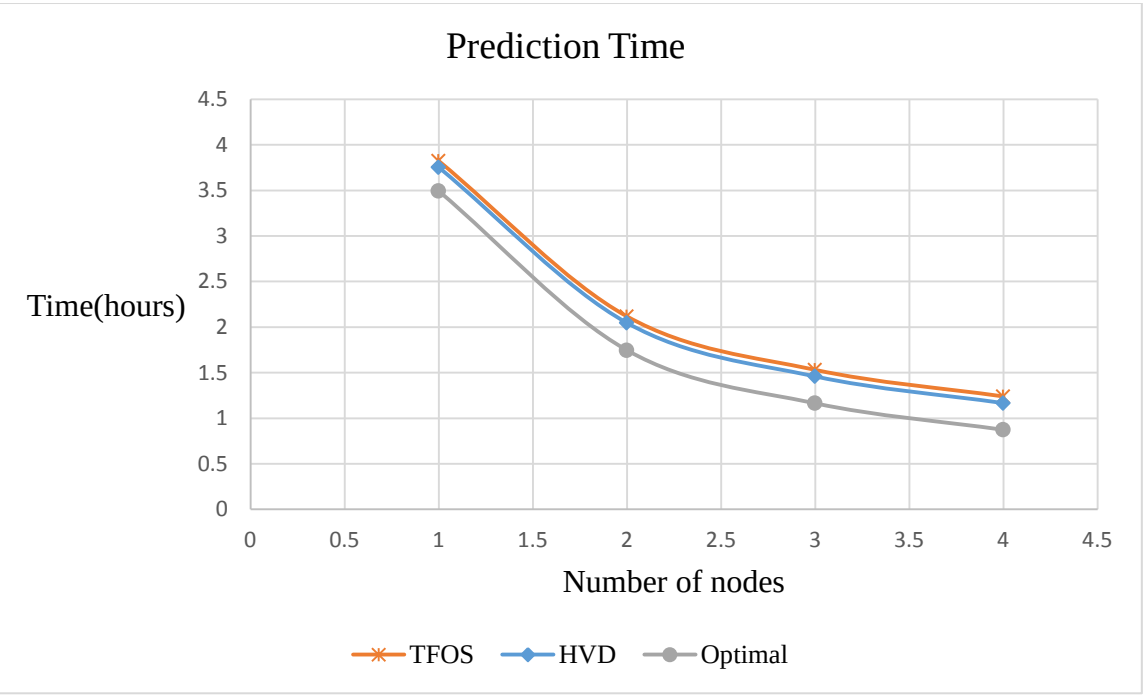
الوضع على معامل توسّع أكبر من 0.81 وهو نسبة مقبولة. أيضا نجد من الشّكل معامل توسّع أفضل ل TFOS بالمقارنة مع HVD ، ويعود السّبب الرّئيسي لذلك لكون زمن التّنفيذ على عقدة واحد باستخدام TFOS هو أقل من زمن التنفيذ على عقدة واحدة باستخدام HVD بمقدار كبير نسبيّاً لأنّ كلفة عمليّات الإعداد وتهيئة عنقود في HVD باستخدام عقدة واحدة أقل من نظيرتها في TFOS، وعند التّنفيذ على أكثر من عقدة يتقارب أداء إطاري العمل، ولكون مقياس التّوسّع يعتمد بالحساب على زمن تنفيذ عقدة واحدة في كل منهما فإنّه من المُتوقّع الحصول على نتائج كالتي حصلنا عليها كما يوضّح الشّكل المذكور آنفاً.



الشكل(8) قابلية توسّع إطاري العمل في تطبيق التّدريب على العنقود مع الحالة المثاليّة بزيادة عدد العقد

أُجريت عمليّة التّنبّؤ على 384 صورة كبيرة الحجم مع زيادة عدد العقد المستخدمة ضمن العنقود بحيث تمّت إعادة التجربة من أجل كل عدد جديد من العقد وحساب الزّمن المُستهلك.

يوضّح الشكل(9) مقارنة بين أداء عمل TensorFlowOnSpark و Horovod مع الحالة المثالية من حيث زمن التّنفيذ عند القيام بعملية التنبّؤ مع زيادة عدد العقد المستخدمة ضمن العنقود بحيث تمّت إعادة التجربة من أجل كل عدد جديد من العقد وحساب الزمّن المُستهلك. وقد استغرق النّظام المُقترح للتنبّؤ ب384 صورة 1 ساعة و10 دقائق باستخدام Horovod و 1 ساعة و14 دقيقة باستخدام TensorFlowOnSpark فيما كان الزّمن المستهلك لتنفيذ نفس التّطبيق وفق منهجيّة الجهاز المنفرد 3 ساعات و36 دقيقة مما يبيّن التّحسين الكبير في أداء النظام.
بملاحظة الشكل(9) نستطيع رؤية أداء مشابه لعمليّة التّدريب لناحية قابليّة التوسّع حيث نرى انخفاضاً في الأداء عند التّنفيذ على عقدة واحدة، ليتوسّع الفارق مع زيادة عدد العقد في العنقود ويستقرّ الوضع عند تكرار التّجربة على عقدتين أو أكثر كما تبيّن الخطوط البيانيّة لكل من الحالة المثاليّة وإطاري العمل المُستخدَمين.



الشكل(9) زمن تنفيذ تطبيق التّنبؤ

6- الخاتمة

تكمن أهمّيّة هذا النّوع من الأبحاث في تخفيض الوقت اللّازم لمعالجة البيانات في تطبيقات الاستشعار عن بعد وخصوصاً تلك التّي تعتمد على تقنيات الذّكاء الصّنعيّ وتخفيض التّكاليف المتعلّقة بتوفير عتاد مُخصّص وباهظ الثمن لمعالجة هكذا أنواع من التطبيقات، وتلك التكاليف كان من الممكن أن تبقى كبيرة بالمقارنة مع النّظم التقليديّة من دون استخدام تقنيات البيانات الكبيرة ونظم الحوسبة فائقة الأداء كالتّي تم استثمارها في هذا البحث لإنجاز العمل المطلوب.

بيّنا في هذا البحث كيف يمكن استثمار تقنيات البيانات الكبيرة ونظم الحوسبة فائقة الأداء في معالجة صور الأقمار الصّنعيّة في تطبيقات التّعلّم العميق المعروفة بحاجتها لموارد ذات قدرات حوسبة عالية أثناء التّنفيذ، هذا وقد واقترح البحث استخدام سبارك كمحرّك أساسي للتنّفيذ ومعالجة البيانات فيما فنّد استخدام

إطـــــاري عمـــــل لأجـــــل المعالجـــــة التّفرعيّـــــة همـــــا Horovod و TenosrFlowOnSpark، وتـــمّ ايضـــاً اســـتُخدام نظـــام الملفّـــات المُـــوزّع الخـــاص بهادووب لتخزين البيانات.

أظهـرت النّتـائج أداء جيّـداً جـدّا للمنهجيّـة المُقترحـة والـنظم الموزعـة علـى منصـات البيانـات الكبيـرة بالمقارنـة لناحيـة زمـن التنفيـذ مـع منهجيّـة الجهـاز المنفـرد فـي كـل مـن تطبيقـيّ التّـدريب والتّنبـؤ باسـتخدام إطـاري العمـل TensorFlowOnSpark و Horovod علـى سـبارك، والأهـمّ مـن ذلـك هـو أنّ الـزمن المُسـتهلك فـي تنفيـذ التطبيقـات باسـتخدام المنهجيّـة المتُبعـة لكـل مـن إطـاري العمـل يقـارب نظيـره فـي الحالـة المثاليـة حيـث أنّ الفـارق بينهمـا كـان مسـتقرا تقريبـا مـع زيـادة حجـم العنقـود. وعنـد مقارنـة إطـاري العمـل تبـيّن لنـا تقـارب الأداء بينهمـا لناحيـة زمـن التّنفيـذ مـع ملاحظـة تفـوّق Horovod فـي التّطبيقـين المـذكورين بفـارق قليـل نسـبيّاً علـى TensorFlowOnSpark بنسـبة تقـارب ال (2.1%) فيمـا كانـت قابليّـة توسّـع TensorFlowOnSpark أفضل بنسبة تقارب ال (5%).

7- التوصيات

إن من الجوانب المهمة والمساهمات الأساسية لدراستنا هو فتح الآفاق لدراسات مستقبلية يمكن العمل عليها من أجل تحسين النتائج وفتح باب اعتماد النموذج المقدم في الدراسة وتطويره ليلائم بيئات العمل المختلفة، وتطبيقه في قطاعات عمل أخرى. ومن هنا نعرض بعض الحالات التي نقترح العمل عليها مستقبلاً:

- اختبار نماذج تعلّم عميق كبيرة لا يمكن استيعابها على جهاز واحد باستخدام منهجيّة تفرّعيّة النّموذج وتقييم أدائها.
- استخدام المنهجيّة المُقترحة في هذا البحث في معالجة تطبيقات ذات طبيعة مختلفة كاكتشاف التغيّرات أو تطبيقات تتطلّب استجابة في الزّمن الحقيقيّ
- لتقنيات الاستشعار عن بعد في مجال تطبيقات معالجة الفيديو الفضائيّ مستقبل واعد للمستثمرين في عدّة مجالات حيث باتت عدّة شركات تقدّم هذه الخدمات،

الّا أنّ هذه التّطبيقات ذات متطلّبات وموارد أعلى بكثير من تلك التّي تعالج الصّور فحسب ومن المهمّ أن يكون المنهج المُقترح قابلاً للتعديل أو التّطوير وقادراً على التّكيّف لتلبية هذه النّوع من مجالات العمل لذلك نوصي باستخدام المنهجيّة المُقترحة في هذه الدّراسة كنواة عمل وتقييم أدائه.

- وضع نمّوذج جاهز للاستثمار كمنتج نهائيّ يمكن للمستثمرين استخدامه في أعمالهم بشكل آلي ويدعم تقديم خدمات المعالجة الأوليّة للبيانات وازالة أو تخفيف أعباء إعداد بيئة العمل المناسبة لتنفيذ التّطبيقات المطلوبة.

المراجع

[1] P. Swarnalatha and P. Sevugan, *Big Data Analytics for Satellite Image Processing and Remote Sensing*. IGI Global, 2018.

[2] H. M. Patel, K. Panchal, P. Chauhan, and M. B. Potdar, "Large scale image processing using distributed and parallel architecture," *Gujrat, India, Int. J. Comput. Sci. Inf. Technol.*, vol. 6, no. 6, pp. 5531–5535, 2015.

[3] R. Yadav and D. M. C. Padma, "Processing of Large Satellite Images using Hadoop Distributed Technology and Mapreduce: A Case of Edge Detection," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 3, no. 5, pp. 3456–3460, 2015.

[4] T. Sharma, V. Shokeen, and S. Mathur, "Distributed Approach to Process Satellite Image Edge Detection on Hadoop Using Artificial Bee Colony," *Int. J. Serv. Sci. Manag. Eng. Technol.*, vol. 11, no. 2, pp. 80–94, 2020.

[5] M. M. U. Rathore, A. Ahmad, A. Paul, and J. Wu, "Real-time continuous feature extraction in large size satellite images," *J. Syst. Archit.*, vol. 64, pp. 122–132, 2016.

[6] B. Shangguan and P. Yue, "SPARK processing of computing-intensive classification of remote sensing images: the case on K-means clustering algorithm," in *2018 26th International Conference on Geoinformatics*, 2018, pp. 1–4.

[7] I. Nurwauziyah, U. Sulistyah, I. Gede, I. G. B. Putra, and M. Firdaus, "Satellite Image Classification using Decision Tree, SVM and k-Nearest Neighbor," *no. July*, 2018.

[8] S. A. Manaf, N. Mustapha, M. Sulaiman, N. A. Husin, and M. R. A. Hamid, "Artificial neural networks for satellite image classification of shoreline extraction for land and water classes of the north west coast of peninsular Malaysia," *Adv. Sci. Lett.*, vol. 24, no. 2, pp. 1382–1387, 2018.

[9] Z. N. Absardi and R. Javidan, "Classification of big satellite images using hadoop clusters for land cover recognition," in *2017*

IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), 2017, pp. 600–603.

[10] J. Dowling, "Distributed TensorFlow," 2017. https://www.oreilly.com/people/jim-dowling/.

[11] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, 2015, pp. 234–241.

[12] A. Foundation, "Apache Spark: Lightning-fast unified analytics engine," 2020. https://spark.apache.org/.

[13] B. Chambers and M. Zaharia, *Spark: The definitive guide: Big data processing made simple*. " O'Reilly Media, Inc.," 2018.

[14] L. Yang, J. Shi, B. Chern, A. Feng, and Y. B. M. Team, "Open Sourcing TensorFlowOnSpark: Distributed Deep Learning on Big-Data Clusters," 2017. https://www.oreilly.com/content/distributed-tensorflow/.

[15] A. Sergeev and M. Del Balso, "Horovod: fast and easy distributed deep learning in TensorFlow," *arXiv Prepr. arXiv1802.05799*, 2018.

[16] J. J. Dai *et al.*, "Bigdl: A distributed deep learning framework for big data," in *Proceedings of the ACM Symposium on Cloud Computing*, 2019, pp. 50–60.

[17] J. Patterson and A. Gibson, *Deep learning: A practitioner's approach*. " O'Reilly Media, Inc.," 2017.

[18] Google, "Introduction to gRPC," 2020. https://grpc.io/docs/what-is-grpc/introduction/.

تحسين أداء البروتوكولات الاستباقية في الشبكات النقالة تلقائية التشكيل باستخدام خوارزميات الذكاء الصنعي

الدكتور مأمون يونس *

عمار محمد غريب **

الملخص

الشبكات النقالة تلقائية التشكيل هي عبارة عن شبكات دون بنية تحتية وهي قابلة للنشر والتكوين الذاتي بسرعة ولا تحتاج إلى دعم مركزي، تتكون الشبكات النقالة تلقائية التشكيل من مجموعة من العُقد المتنقلة التي تعمل كجهاز توجيه ومضيف في نفس الوقت و تكون قادرة على نقل حركة المرور من عقدة إلى أخرى ، وتتحرك العُقد في الشبكة بسرعة وحركة عشوائية مما يسبب تغير مستمر في طوبولوجيا الشبكة.

إن مسألة التوجيه في الشبكة واختيار المسار الأفضل بين العقد من أكثر الأمور التي تجذب انتباه الباحثين في مجال الشبكات النقالة وذلك بسبب أهمية عملية التوجيه وتأثيرها على أداء الشبكة وتشكل عملية التوجيه تحدياً كبيراً نتيجة محدودية التقنيات اللاسلكية المستخدمة وتغير الطوبولوجيا بشكل كبير نتيجة حركة العقد.

تم التركيز في هذا البحث على تحسين أداء بروتوكول OLSR الاستباقي من أجل اختيار المسار الأفضل للتوجيه الذي يحقق أقل نسبة تأخير زمني في الشبكة ويؤمّن أفضل نسبة تسليم للرزم خلال عملية الإرسال. تم استخدام خوارزمية مستعمرة النمل من أجل اختيار أفضل مسار بالاعتماد على عاملين أساسيين لتقييم المسارات وهما طول المسار و إنشغالية العقد الموجودة ضمن المسار، حيث تم بناء شبكة الشبكات النقالة تلقائية التشكيل باستخدام

* أستاذ مساعد – قسم هندسة الحواسيب و الأتمتة – كلية الهندسة الميكانيكية والكهربائية – جامعة دمشق – دمشق – سورية .

** طالب دراسات عليا (ماجستير)– هندسة الحواسيب وشبكاتها – كلية الهندسة الميكانيكية والكهربائية – جامعة دمشق – دمشق – سورية .

المحاكي NS2.35 وتم تنفيذ عدة سيناريوهات لاختبار أداء البروتوكول المحسن من حيث زيادة عدد العقد المتحركة في الشبكة وزيادة سرعة العقد المتحركة في الشبكة ، وقد أظهرت نتائج الاختبار تقليل التأخير الزمني في الشبكة وزيادة نسبة تسليم الرزم.

الكلمات المفتاحية :

الشبكات النقالة تلقائية التشكيل , بروتوكولات التوجيه , بروتوكول حالة الوصل المحسن , خوارزمية مستعمرة النمل , NS2.35.

Improving the Performance of Proactive Protocols in Manets Networks Using Artificial Intelligence Algorithms

Dr Mamoun Younes [*]
Ammar Mohammad Ghareeb [**]

Abstract

Mobile Ad-Hoc Networks (MANETs) are infrastructure-less networks that are rapidly deployable and self-configuring and do not need central support. MANETs consist of a group of mobile nodes that act either as a router or as a host. Nodes in these network move rapidly and randomly, causing a continuous change in network topology.

The routing in the network and choosing the best path between nodes are major issues that attract the attention of researchers in the field of mobile networks, because of the importance of the routing process and its impact on network performance. This paper focuses on

[*] **Assistant Professor - Department of Computer and Automation Engineering - Faculty of Mechanical and Electrical Engineering - University of Damascus - Damascus - Syria.**
[**] **Postgraduate student (Master) - Computer and Network Engineering - Faculty of Mechanical and Electrical Engineering - Damascus University - Damascus - Syria.**

improving the performance of the proactive OLSR protocol in order to choose the best routing path that achieves the least time delay in the network, secures the best packet delivery rate and ensures reducing packet loss during the transmission process. The ant colony algorithm was used to choose the best path based on two main factors , namely the path length and the occupancy of the nodes within the path. Our simulation scenarios are built using NS2.35 to test the performance of the improved protocol in terms of increasing the number of nodes in the network and increasing the speed of nodes in the network. The test results show a reduction in the time delay in the network and an increase in the packet delivery rate.

Keywords:

Manets , OLSR , ACO , Routing Protocol , NS2.35 , PDR

1- مقدمة

الشبكات النقالة تلقائية التشكيل (Mobile Ad-hoc Networks) هي مجموعة من العقد اللاسلكية (حاسب محمول ، هاتف نقال ، إلخ) لديها القدرة على التواصل مع بعضها البعض دون أي اعتماد على بنية أساسية داعمة ثابتة أو إدارة مركزية.

لذلك فإن MANET هي شبكة تلقائية تظهر تلقائيًا عندما تتجمع العقد معًا ، يمكن للعقد في MANET التواصل مع جميع العقد الأخرى ضمن نطاق الراديو الخاص بها أو يمكنها استخدام العقد الوسيطة للتواصل مع العقد غير الموجودة في نطاق الراديو الخاص بها [1].

يتم نقل المعلومات من المصدر إلى الوجهة عبر العقد الوسيطة بحيث تعمل كل عقدة في الشبكة كمضيف وجهاز توجيه، وتتميز الشبكات MANET بطوبولوجيا ديناميكية متغيرة باستمرار ، وموارد مقيدة.

إن السمتان الرئيسيتان للشبكات النقالة هما التنقل والاتصال متعدد القفزات بين العقد، كما تميل العقد إلى التحرك بحرية في أي اتجاه وفي أي وقت ، وبالتالي تقوم في كثير من الأحيان بإنشاء أو قطع الروابط مع العقد الأخرى.

يتم التواصل بين العقد في الشبكة عن طريق مجموعة من بروتوكولات التوجيه التي تضبط عملية نقل البيانات بين المصدر والهدف، ويكمن التحدي الأكبر في هذا النوع من الشبكات في إيجاد مسارات التوجيه الأكثر فعالية مع مراعاة التغير في طوبولوجيا الشبكة واستنزاف طاقة العقد والتأخير الزمني الحاصل في الشبكة [2].

2- هدف البحث

إن مسألة التوجيه في شبكات الشبكات النقالة تلقائية التشكيل هي من أهم الأمور في الشبكات [3] ، حيث تشكل مسألة الحصول على المسار الأمثل ضمن الشبكة تحدياً كبيراً للباحثين ، لذلك سنقوم في هذا البحث بإجراء تحسين لبروتوكول OLSR من أجل اختيار أقصر وأفضل مسار ضمن مسارات التوجيه بين العقدة المصدر و العقدة الهدف، حيث يركز البحث على تقليل التأخير الزمني ضمن الشبكة وعلى تحقيق أعلى نسبة تسليم للرزم بشكل سليم بين

العقد، وسيتم ذلك من خلال استخدام خوارزمية أمثلة لبناء مسارات التوجيه والمفاضلة بينها لاختيار المسار الأفضل حسب التابع الهدف المرغوب.

3- الأسس النظرية للبحث

3-1- بروتوكولات التوجيه في الشبكات النقالة تلقائية التشكيل (Manets)

التوجيه هو عملية تحديد المسار في شبكة يتم من خلالها إرسال الحزم الموجهة من العقدة المصدر إلى العقدة الوجهة النهائية الخاصة بها من خلال العقد الوسيطة. يتضمن التوجيه بشكل أساسي مكونين هما تحديد مسار التوجيه الأمثل ونقل الحزم عبر المسار الذي تم تحديده.

إن تحديد المسار الأمثل هو مشكلة معقدة. تم تصميم بروتوكول التوجيه لتحديد كيفية اتصال العقد المتنقلة مع بعضها البعض لنشر المعلومات ، تستخدم بروتوكولات التوجيه العديد من المقاييس مثل طول المسار والموثوقية وتأخير التوجيه وعرض النطاق الترددي والحمل وتكاليف الاتصال وما إلى ذلك لتقييم أفضل مسار للتوجيه من بين العديد من البدائل لتوجيه حزم البيانات [4].

3-2- بروتوكولات التوجيه الاستباقية (Proactive Routing Protocols)

يتم تشكيل المسارات إلى كل العقد في الشبكة بشكل مسبق وتقوم العقد في الشبكة بشكل دوري بإرسال رسائل لتحديث معلومات التوجيه و تحتفظ كل عقدة بمعلومات التوجيه إلى كل عقدة أخرى (أو عقد موجودة في جزء معين) في الشبكة ، و عادة ما يتم الاحتفاظ بمعلومات التوجيه في عدد من الجداول المختلفة. يتم تحديث هذه الجداول بشكل دوري و / أو إذا تغير هيكل الشبكة [5].

3-2-1- بروتوكول توجيه حالة الربط المحسن Optimized Link State Routing Protocol (OLSR)

هو بروتوكول توجيه استباقي لحالة الارتباط يحاول تقليل عدد حزم معلومات التوجيه وحجمها من خلال اختيار عقدة شبكة جديدة وتسميتها بـ Multipoint Relays (MPRs).

تمكنت هذه العناصر الجديدة من تجنب عمليات إعادة الإرسال الزائدة عن الحاجة وكمية معلومات حالة الارتباط الواردة في رسائل التحديث ، كما يمكن رؤيته في الفصل التالي المخصص بشكل خاص لـ OLSR.

,نظرًا لطبيعة البروتوكول الاستباقية ، يعد OLSR أسرع من البروتوكولات التفاعلية لأنه يحتوي على المسارات المتاحة على الفور عند الحاجة ، لذلك يتعين عليه فقط معرفة أفضل طريق إلى الوجهة في جدول التوجيه الخاص به.

هذا البروتوكول هو البروتوكول الأمثل عند العمل على شبكات كبيرة وكثيفة ، لأن التحسين الذي تقدمه المرحلات متعددة النقاط (Multipoint Relays MPRs) يكون أعلى في هذا السياق وليس في الشبكات الأصغر[6].

ميزة أخرى مهمة لبروتوكول OLSR هي أنه لا يعتمد على أي عنصر مركزي ، لذلك فهو يعمل بطريقة موزعة بالكامل وهي الأمثل في الشبكات المخصصة. بالإضافة إلى ذلك ، لا يحتاج البروتوكول إلى القلق بشأن ترتيب تسليم الحزم لأن كل رسالة تحكم تحتوي على رقم تسلسلي. بهذه الطريقة ، يمكن للطرف المستلم تحديد الحزمة التي تحتوي على أحدث المعلومات ويمكنه تجاهل الحزمة القديمة للحفاظ على معلومات جديدة حول هيكل الشبكة.

3-2-2- رسائل التحكم ضمن بروتوكول (OLSR)

يعتمد عمل بروتوكول OLSR على رسالتي تحكم أساسيتين:

1- HELLO messages

2- Topology Control (TC)

يتم إرسال رسائل HELLO بشكل دوري بواسطة جميع عقد الشبكة وتستخدم لاستشعار الارتباط واكتشاف الجوار وتحديد اختيار MPR. من ناحية أخرى، يتم إرسال رسائل TC بواسطة MPRs وتحتوي على معلومات الهيكل التي تستخدمها العقد لحساب جداول التوجيه الخاصة بها.

3-3- خوارزمية مستعمرة النمل (Ant Colony Optimization)

هي عبارة عن تقنية احتمالية تقوم بالبحث عن المسار الأمثل في الرسم البياني بالاعتماد على سلوك النمل في بحثه عن طريق بين مستعمرته لإيجاد مصدر الغذاء. تعتبر خوارزمية مستعمرات النمل (The ant colony optimization algorithm ACO) من خوارزميات البحث التي تعتمد على التجربة والخطأ والتي تعطي حل مقبول (قد يكون افضل حل وقد لا يكون) لذلك يتم استخدامها في حل المسائل التي تأخذ وقت طويل باستخدام الحاسوب مثال ذلك مسائل NP-Complete (او المسائل التي تحتاج الى تجربة كل الاحتمالات حتى تصل الى الحل المطلوب ان وجد) [7].

اتت فكرة الخوارزمية من محاكاة عملية البحث عن الطعام عند النمل وهي كالتالي:

1- تقوم مجموعة من النمل بالانطلاق من الخلية في عدة اتجاهات عشوائية (هذه العملية تتم في المرة الاولى فقط في المرات اللاحقة يتم اختبار كل مسار واختيار مسار معين كما سنرى لاحقا).

2- اثناء مرورها تقوم النملة بإفراز مادة تسمى فيرمون بنسبة معينة (فائدة هذه المادة معرفة الطريق الذي مرت فيه)

3- عندما تجد مصدر للطعام فهي تأخذ كمية منه وتعود الى الخلية عن طريق اختيار مسار معين (المسار الذي يحوي اكبر كمية فيرمون).ايضا عند عودتها ستقوم بإفراز نفس الكمية من الفيرمون.

4- عندما تنطلق النملة من الخلية مجددا ستقوم باختبار كمية الفيرمون في كل مسار وتختار المسار الذي يحوي اكبر كمية من الفيرمون.

5- يتم تحديث كمية الفيرمون كل فترة زمنية معينه (تركيز الفيرمون يتلاشى بمرور الوقت).

3-3-1- محاسن خوارزمية النمل

- ملاحظات ردود ايجابية لاكتشاف سريع للحلول الجيدة
- فعالة لمشكلة البائع المتجول وما شابه ذلك من مشاكل يمكن استخدامها في التطبيقات الديناميكية (تتكيف مع التغييرات مثل المسافات الجديدة ، وما إلى ذلك)
- وراثة التفرع Inherent parallelism

3-3-2- مساوئ خوارزمية النمل

- التحليل النظري صعب
- تسلسلات القرارات العشوائية (غير مستقلة)
- تغير توزيع الاحتمالات بالتكرار

4- طرائق البحث ومواده

4-1- تحسين بروتوكول OLSR باستخدام خوارزمية مستعمرة النمل (ACO)

قام العديد من الباحثين بالعمل على تحسين بروتوكول OLSR باستخدام خوارزميات أمثلة عديدة ومتنوعة وتناولوا عدة معاملات للدراسة والتحسين. سنقوم في هذا البحث باستخدام خوارزمية ACO في عملية التحسين، وسنعمل في هذا البحث على اختيار أفضل مسار من مسارات التوجيه المتاحة في الشبكة بين العقدة المصدر والعقدة الهدف، وذلك بالاعتماد على مفهومين أساسيين هما طول المسار وانشغالية العقد، حيث أن طول المسار هو عدد العقد الموجودة على هذا المسار وكلما زاد عدد العقد في المسار كلما زاد التأخير في توصيل رزم المعلومات من المصدر إلى الوجهة، وإن انشغالية العقد تدل على أن العقدة تحوي مجموعة رزم من المعلومات تقوم بمعالجتها وبالتالي يجب على البروتوكول انتظار العقدة حتى تنتهي من عملية معالجة رزم المعلومات مما يؤدي إلى حدوث تأخير في زمن توصيل الرزم ضمن الشبكة.

ستتضمن آلية العمل الخطوات التالية من أجل الوصول إلى التحسين المطلوب:

1- يبدأ بروتوكول OLSR بالعمل ويتم اختيار عقد الـ MPR بالنسبة لكل عقدة ويتم تشكيل جداول التوجيه بالاعتماد على معلومات طوبولوجيا الشبكة وعلى عُقد الـ MPR التي تم اختيارها.
2- يتم تهيئة بارامترات خوارزمية مستعمرة النمل ACO بالقيم المطلوبة.
3- تقوم الخوارزمية بالحصول على معلومات طوبولوجيا الشبكة وعُقد الـ MPR ومسارات التوجيه المتاحة من البروتوكول.
4- تعمل الخوارزمية على اختيار المسار الأفضل من المسارات المتاحة بناءً على معاملات طول المسار وانشغالية العقد من أجل الوصول إلى أفضل مسار توجيه ضمن المسارات المتاحة في جداول توجيه البروتوكول.

4-2- خطوات عمل الخوارزمية لاختيار مسار التوجيه الأفضل

تم ضبط بارامترات الخوارزمية اعتماداً على [8] وبما يناسب التحسين المقترح في البحث وفق ما يلي :

1- تهيئة بارامترات الخوارزمية.
2- تحديد العقدة المصدر S والعقدة الهدف D ، من أجل حساب المسار الأفضل بين المصدر والهدف.
3- تبدأ العقدة المصدر S بإرسال (Path_Request_Ant) إلى الوجهة الهدف من خلال جميع جيرانها الذين يقعون على مسافة قفزة واحدة من S، سيتم تحديد القفزة التالية وفق أعلى احتمال للارتباط كما هو مبين في المعادلة (1):

$$P_{i,j} = [\tau_{ij}(t)]^{\alpha} / \sum_{k \in allowed_k} [\tau_{ik}(t)]^{\alpha}$$

حيث:

$\tau(i, j)$: قيمة الفورومون في العقدة.

α : بارامتر تحكم بتأثير الفوروموون

تحتفظ كل عقدة ضمن الشبكة بجدول يدعى جدول الفوروموون "PheroTable" يحتوي على قيمة الفوروموون المتاحة على كل رابط والتي تتم تهيئتها إلى ثابت C. كما تحتفظ كل عقدة بجدول احتمالية "ProbTable" يحتوي على احتمال الانتقال لتحديد العقد المجاورة.

4- تحديث قيمة الفوروموون في جدول الفوروموون للعقدة وفق المعادلة (2):

$$\tau_{i,j} = (1 - \rho)\tau_{i,j} + \Delta\tau_{i,j}$$

حيث:

$\boldsymbol{\rho}$: معدل تبخر الفيرمون

$\boldsymbol{\Delta\tau_{i,j}}$: هي كمية الفيرمون المخزنة

5- عندما يصل Path_Request_Ant إلى الوجهة ، سيتم تحويله إلى Path_Reply_Ant وإعادة توجيهه نحو عقدة المصدر الأصلية ، واتخاذ نفس المسار المقابل لـ Path_Request_Ant ولكن في اتجاه عكسي.

6- يتم تحديث قيم الفوروموون أثناء المرور في المسار العكسي وفق المعادلة (2)

7- اختيار المسار الأمثل وفق قيم الفوروموون في المسار وحسب التابع الهدف التالي:

$$\text{Total Route Fitness} \quad (\text{TRF}) = \tau + 1/(\alpha_1 N_{hob} + \alpha_2 T_{hold})$$

حيث :

N_{hob} : *عدد القفزات ضمن المسار*

T_{hold} : *زمن الانتظار في العقدة*

α_1 ، α_2 : *بارامترات يتم ضبطها بحيث يكون* $\alpha_2 + \alpha_1 = 1$

5- بيئة المحاكاة

تم في هذا البحث استخدام محاكي الشبكات (Network Simulation 2) NS2، وهو عبارة عن محاكي يستخدم مفهوم الأحداث المتقطعة والتي تعتبر واحدة من طرق المحاكاة ويحوي المحاكي على العديد من مكونات وبروتوكولات الشبكات ، حيث يستخدم هذا المحاكي لغة Otcl من أجل كتابة سيناريو المحاكاة ومسارات التحكم وتهيئة الشبكة كما يستخدم لغة C++ من أجل كتابة مسارات البيانات ومعالجة الرزم وتوليد المحاكاة [9].

وتم كتابة ملفات برمجية بلغة AWK من أجل تحليل ملف مراقبة الشبكة واستخلاص النتائج منه [10].

6- الإجراء العملي ومناقشة النتائج

قمنا باستدعاء بروتوكول OLSR وتطبيق خوارزمية مستعمرة النمل ACO من أجل اختيار المسار الأفضل بالاعتماد على تابع الهدف وتابع الاحتمال للخوارزمية ، وقبل البدء بالتنفيذ يجب ضبط مجموعة من البارامترات من أجل إنشاء الشبكة وضمان عملها بشكل صحيح إذ تم استخدام برنامج المحاكاة (Network Simulator 2.35) الذي تم تنصيبه على نظام تشغيل (Ubuntu 16.04) وتم تحديد بيئة العمل التي ستنتشر ضمنها العقد بمساحة (1000m X 1000m) وتم اختيار البروتوكول TCP كبروتوكول نقل ضمن الشبكة ، كما تم اختيار البروتوكول (IEEE 802.11) من أجل اتصال العقد فيما بينها وتم تزويد العقد بطاقة ابتدائية مقدارها (J 150) ، كما تم استخدام بروتوكول نقل الملفات FTP من أجل توليد ونقل الرزم بين العقد ، وتم اختيار حركة عشوائية للعقد من أجل محاكاة السلوك الطبيعي للعقد ضمن الشبكة ، وإن عدد العقد في الشبكة وسرعة العقد يتغير حسب السيناريو الذي يتم تطبيقه ، ويتم توضيح البارامترات في الجدول التالي:

الجدول (6-1) بارامترات الشبكة النقالة في السيناريو الأول

Parameters Name	Description
Network Simulator	NS 2.35
Simulation Environment	Ubuntu 16.04
Environment Size	1000m X 1000m
Routing Protocol	OLSR
Number of Mobile Nodes	20 , 30 , 40 , 50 , 60 , 70 , 80
Transport Protocol	TCP
MAC Protocol	IEEE 802.11
Mobility Model	Random Motion Model
Traffic Type	FTP
Simulation Time	100 seconds
Nodes Speed	5 m/s
Initial Energy of Node	150 **J**

وتم ضبط بارامترات خوارزمية الأمثلة لتحقق التحسين المطلوب وفق ما يلي:

الجدول (6-2) بارامترات خوارزمية مستعمرة النمل

البارامتر	الرمز	القيمة
عدد النمل	k	20
عدد التكرارات	n	40
بارامتر التحكم بتأثير الفوروموون	α	0.3
معدل تبخر الفيرمون	ρ	0.1
معدل الفيرمون الابتدائي في العقد	τ_0	0.1

وتم اختيار تابع هدف من أجل اختيار أفضل مسار من حيث طول المسار وانشغالية العقد ضمن المسار :

- التابع الهدف : $\mathbf{TRF} = \boldsymbol{\tau} + \mathbf{1}/(\propto_{\mathbf{1}} \boldsymbol{N_{hob}} + \propto_{\mathbf{2}} \boldsymbol{T_{hold}})$

حيث أن : N_{hob} : هي عدد القفزات للعقدة

T_{hold} : هي زمن انتظار العقدة حتى تصبح متاحة.

τ : قيمة الفوروموون على المسار

تم العمل على سيناريوين مختلفين من أجل اختبار أداء البروتوكول بعد التحسين حيث تم اختبار البروتوكول بالنسبة لتغير عدد العقد في الشبكة وبالنسبة لتغير سرعة العقد.
وتم اختبار أداء الشبكة والتحسين المقترح لمقارنة البارامترات التالية للشبكة الناتجة عن تنفيذ السيناريوين.

6-1- بارامترات الأداء للشبكة

تم اعتماد نفس البارامترات الواردة في [11] وهي :

6-1-1- التأخير (Delay)

تأخير نهاية لنهاية للرزمة (E2E) هو الزمن المستغرق من توليد الرزمة من قبل المصدر حتى استقبالها من قبل الوجهة، بالتالي هو الزمن الذي تستغرقه الرزمة لعبورها للشبكة ويقدر بالثانية ، وهكذا تدعى جميع التأخيرات في الشبكة بتأخير نهاية لنهاية للرزمة (packet end-to-end) ، ويعطى بالعلاقة :

$$\boldsymbol{T_s - T_r} = \mathbf{D}$$

حيث : T_s : زمن إرسال الرزمة من العقدة المصدر .

T_r : زمن إستقبال الرزمة في الغقدة الهدف .

6-1-2- نسبة تسليم الرزم (PDR)

هي النسبة بين عدد الرزم التي تم استقبالها من قبل العقدة الهدف إلى عدد الرزم التي تم إرسالها من العقدة المصدر، وتعطى بالعلاقة :

$$\mathbf{PDR} = \frac{\sum_{i=1}^{n} \textbf{Total packets received by all destinations}}{\sum_{i=1}^{n} \textbf{Total packets sent by all sources}}$$

6-1-3- الفائض (Overhead)

هو عبارة عن نسبة عدد رزم التحكم المرسلة في الشبكة لاستكشاف مسارات التوجيه إلى عدد الرزم المستقبلة.

6-2- سيناريوهات عمل الشبكة

6-2-1- السيناريو الأول

في السيناريو الأول تم اختبار أداء البروتوكول المحسن من أجل عدد مختلف من العقد في الشبكة حيث تم تنفيذ ثمانية تجارب بحيث تشمل زيادة عدد العقد (20 ، 30 ، 40 ، 50 ، 60 ، 70 ، 80) ، وبعد استخلاص النتائج تم قياس قيم التأخير الزمني (Delay) ونسبة تسليم الرزم (PDR) ونسبة الفائض (Overhead) ، وتم المقارنة بين النتائج قبل عملية التحسين وبعدها.

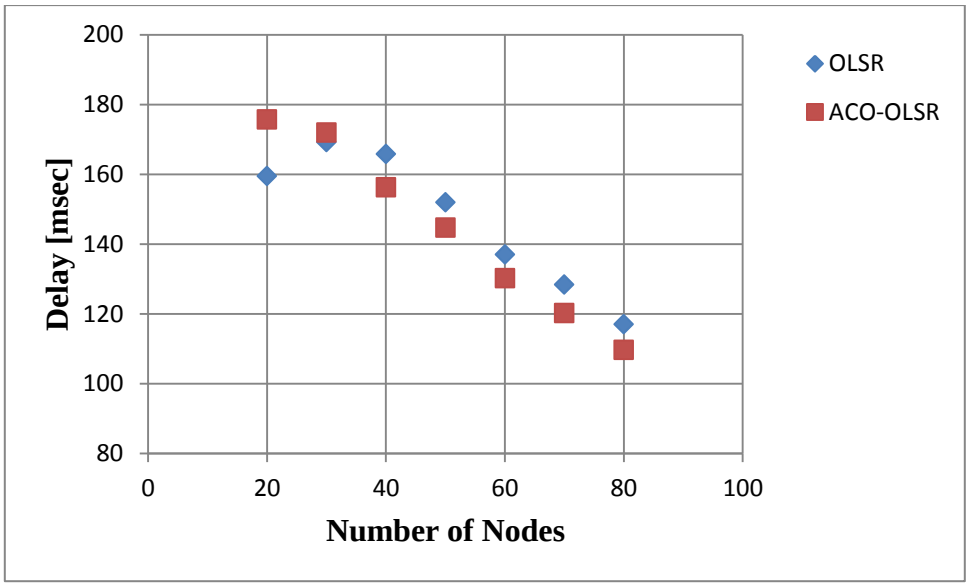
بعد تنفيذ السيناريو الأول تم الحصول على نتائج أداء البروتوكول المحسن الموضحة في الجدول (6 – 3) وتم مقارنة هذه النتائج مع نتائج أداء البروتوكول قبل عملية التحسين وقمنا برسم الخطوط البيانية التي توضح عملية المقارنة.

الجدول (6 – 3) نتائج أداء البروتوكول المحسن من أجل السيناريو الأول

Number of nodes	DELAY%			PDR%			Overhead%		
	OLSR	ACO-OLSR	Improvement Ratio	OLSR	ACO-OLSR	Improvement Ratio	OLSR	ACO-OLSR	Improvement Ratio
20	159.46	175.72	-10.19%	99.28	99.06	-0.22%	269.84	372.18	37.92%
30	169.23	171.97	-1.61%	98.93	99.38	0.45%	340.2	415.39	22.10%
40	165.83	156.22	5.79%	98.69	99.34	0.65%	380.58	466.12	22.47%
50	151.91	144.71	4.73%	98.64	99.13	0.49%	404.7	475.15	17.40%
60	136.98	130.21	4.93%	98.62	99.11	0.49%	449.87	663.72	47.53%
70	128.42	120.27	6.35%	98.63	99.03	0.40%	490.8	692.2	41.03%
80	117.05	109.69	6.28%	98.61	98.99	0.38%	556.11	774.47	39.26%
Average	146.98	144.11	2.32%	98.77	99.14	0.38%	413.15	551.31	32.53%

ومن الجدول (6 – 3) بمقارنة قيم التأخير الزمني قبل عملية تحسين البروتوكول وبعدها نجد أنه عندما يكون عدد العقد في الشبكة قليل يكون أداء البروتوكول قبل عملية التحسين أفضل من أدائه بعدها ، وذلك لأن عدد المسارات في الشبكة قليل ولا داعي لاستخدام عمليات معالجة إضافية تستغرق المزيد من الوقت والتكرارات لاختيار المسار الأفضل ، أما عندما يصبح عدد العقد في الشبكة كبير يصبح من الصعب على البروتوكول إيجاد المسار الأفضل نتيجة حركة العقد وتغير المسارات باستمرار مما ينتج عنه زيادة في التأخير الزمني نتيجة عدم اختيار المسار الأفضل ، ونستنتج أيضاً من الجدول أنه بزيادة عدد العقد في الشبكة بشكل عام ينخفض التأخير الزمني وذلك لأنه بزيادة عدد العقد يزداد عدد المسارات المتاحة

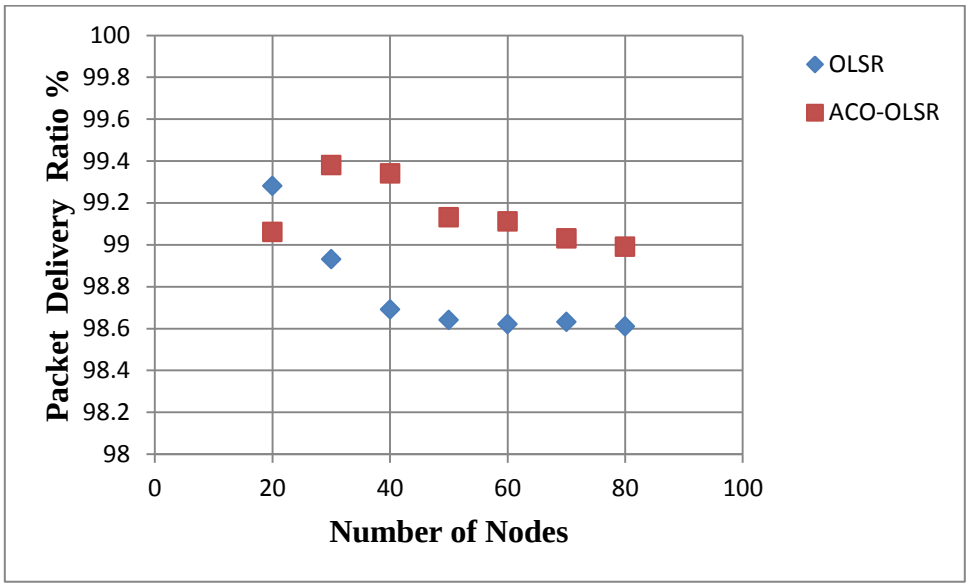
للإرسال وبالتالي ممكن اختيار أفضل مسار يحقق أقل تأخير زمني ، وتظهر النتائج أن البروتوكول الذي تم تحسينه يحقق تأخير زمني أقل بنسبة 2.32% من البروتوكول قبل التحسين ، يوضح الشكل (6 – 1) هذه المقارنة:



الشكل (6 – 1) مقارنة التأخير الزمني بين البروتوكول قبل عملية التحسين وبعدها عند زيادة عدد العقد

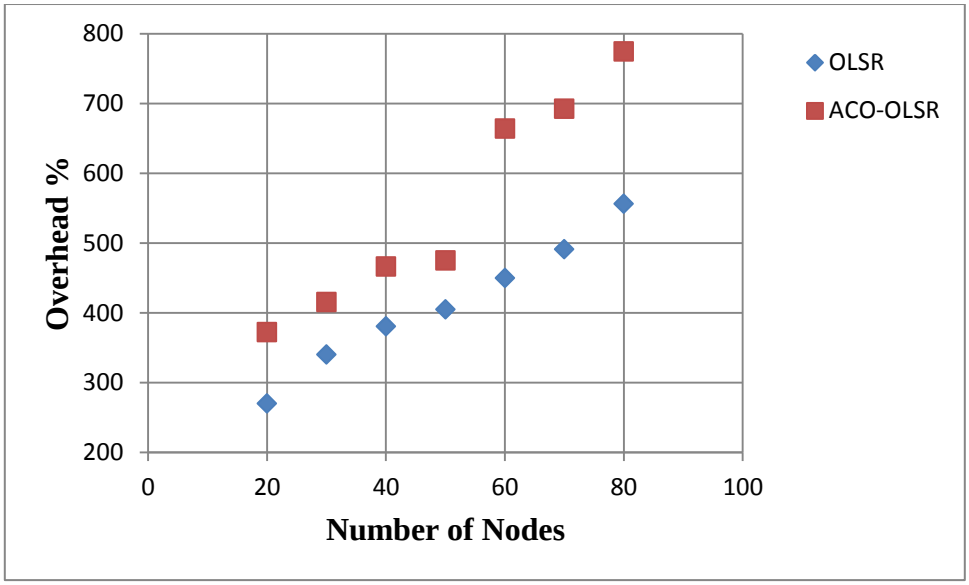
وبعد مقارنة نسبة تسليم الرزم PDR بين البروتوكولين قبل عملية التحسين وبعدها ، نجد أن البروتوكول الذي تم تحسينه يحقق نسبة PDR أعلى بمقدار 0.38% من البروتوكول قبل عملية التحسين ، وذلك لأنه عند استخدام البروتوكول المحُسن نتمكن من اختيار المسار الأفضل وبالتالي نحصل على مسارات أكثر وثوقيه ونتجنب انهيار المسار الناتج عن تغير موضع العقد بسبب حركتها العشوائية ، وبالتالي نضمن وصول الرزم من المصدر إلى الهدف مما يحقق نسبة تسليم رزم (PDR) أعلى من قبل. كما نستنتج أنه بزيادة عدد العقد في الشبكة تزداد نسبة تسليم الرزم (PDR) وذلك لأنه مع زيادة عدد العقد يزداد عدد المسارات المتاحة وبالتالي ينخفض احتمال اختيار مسار سيء أو اختيار مسار من الممكن أن ينهار

لسبب ما وهذا يؤدي إلى زيادة نسبة تسليم الرزم ، وفي المقابل عندما يكون عدد العُقد في الشبكة قليل من الممكن أن ، ويوضح الشكل (6 – 2) هذه المقارنة :



الشكل (6 – 2) مقارنة نسبة PDR بين البروتوكول قبل وبعد عملية التحسين عند زيادة عدد العقد

وعند مقارنة نسبة الفائض (Overhead) بين البروتوكول قبل عملية التحسين وبعدها ، نجد أن بعد تطبيق التحسين على البروتوكول تزداد نسبة الفائض (Overhead) في الشبكة بمقدار 32.53% وذلك بسبب زيادة عدد رسائل التحكم المرسلة من أجل اكتشاف أفضل مسار وبسبب تكرار عملية الإرسال ضمن الخوارزمية حتى نصل إلى المسار الأفضل مما ينتج عنه زيادة الفائض في الشبكة بشكل عام ، ويوضح الشكل (6 – 3) نتائج هذه المقارنة :



الشكل (6 – 3) مقارنة نسبة Overhead بين البروتوكول قبل عملية التحسين وبعدها عند زيادة عدد العقد

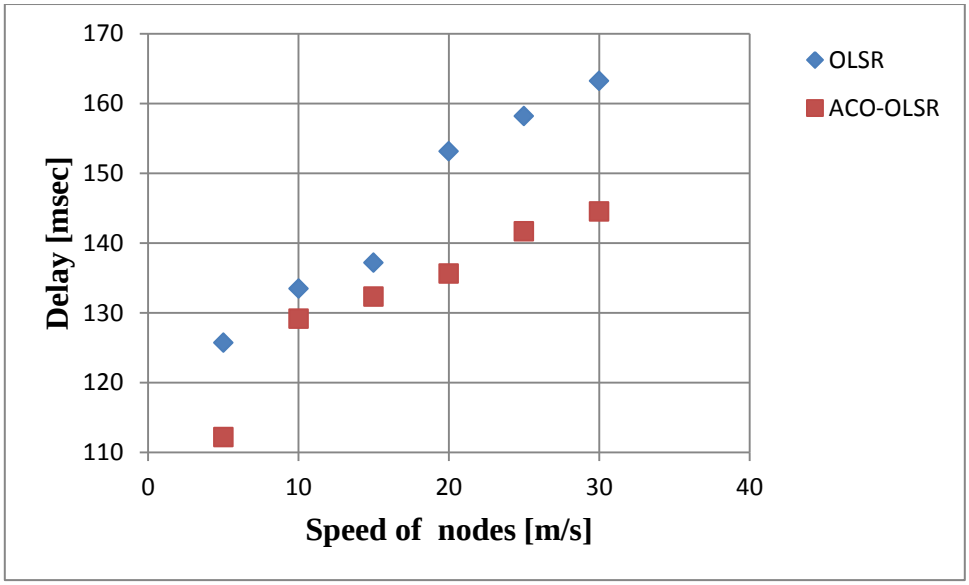
6-2-2- السيناريو الثاني

في السيناريو الثاني تم تنفيذ ستة تجارب من أجل اختبار أداء البروتوكول في حالة سرعات مختلفة للعقد المتحركة (5 ، 10 ، 15 ، 20 ، 25 ، 30) متر بالثانية وتم اختيار عدد عقد ثابت في الشبكة وهو 40 عقدة في جميع الحالات ، وتم استخلاص النتائج الموضحة في الجدول (6- 4) و دراسة تأثير تغير سرعة العقد على أداء البروتوكول، كما قمنا بقياس قيم التأخير الزمني ونسبة تسليم الرزم والفائض الناتجين عن تطبيق البروتوكول المحسن ومقارنتها مع القيم الناتجة عن تطبيق البروتوكول قبل تطبيق التحسين المقترح ضمن البحث.

الجدول (6 – 4) نتائج أداء البروتوكول المحسن من أجل السيناريو الثاني

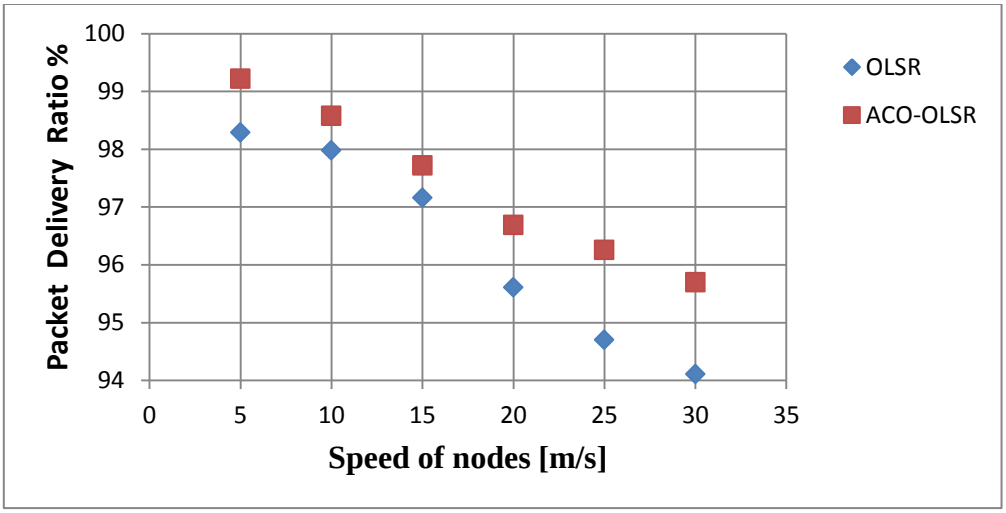
Nodes Speed [m/s]	Delay%			PDR%			Overhead%		
	OLSR	ACO - OLSR	Improvement Ratio	OLSR	ACO - OLSR	Improvement Ratio	OLSR	ACO - OLSR	Improvement Ratio
5	125.71	112.19	10.75 %	98.29	99.22	0.94%	325.4	445.73	36.97 %
10	133.47	129.13	3.25%	97.98	98.58	0.61%	336.69	463.04	37.52 %
15	137.16	132.31	3.53%	97.16	97.72	0.57%	358.39	472.4	31.81 %
20	153.12	135.63	11.42 %	95.61	96.69	1.12%	374.87	529.71	41.30 %
25	158.17	141.65	10.44 %	94.7	96.26	1.64%	450.16	568.16	26.21 %
30	163.21	144.51	11.45 %	94.11	95.7	1.68%	493.84	795.34	61.05 %
Average	145.14	132.57	8.66%	96.30	97.36	1.10%	389.89	545.73	39.96 %

ومن الجدول (6–4) بمقارنة قيم التأخير الزمني في الشبكة للبروتوكول قبل عملية التحسين وبعدها نجد أن البروتوكول بعد عملية التحسين يحقق تأخير زمني أقل بنسبة 8.66% من البروتوكول قبل التحسين ، ونلاحظ أن التأخير الزمني يزداد مع زيادة سرعة العقد المتحركة في الشبكة وذلك بسبب فشل أو انقطاع مسارات التوجيه نتيجة زيادة سرعة العقد أو انهيار بعض العقد بسبب نفاذ طاقتها نتيجةً لزيادة سرعة الحركة مما يسبب انتظار أكبر للرزم في الرتل حتى تصل إلى الهدف وهذا ما ينتج عنه زيادة في التأخير الزمني ، ويوضح الشكل (6 – 4) هذه المقارنة:



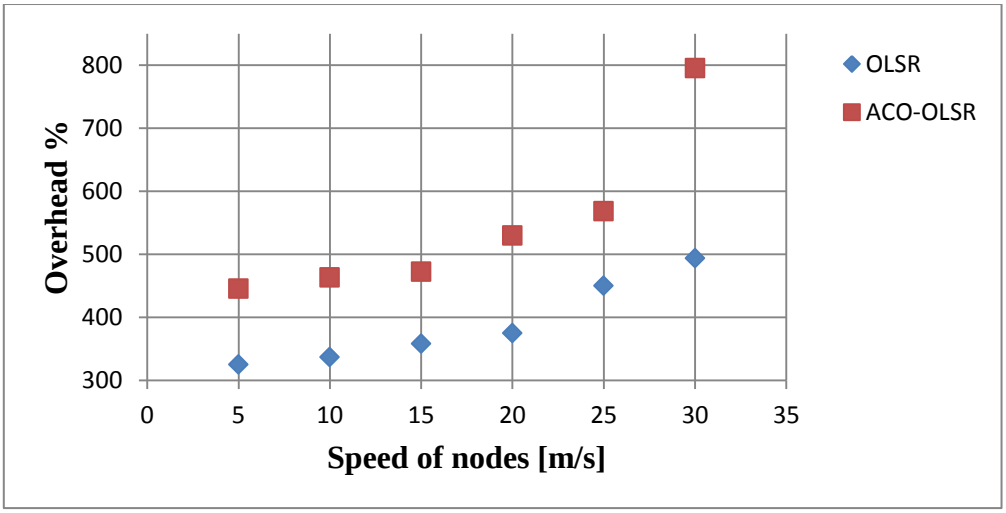
الشكل (6 – 4) مقارنة التأخير الزمني بين البروتوكول قبل عملية التحسين وبعدها عند زيادة سرعة العقد المتحركة

كما نجد من الجدول (6–4) بعد مقارنة نسبة تسليم الرزم (PDR) للبروتوكول قبل عملية التحسين وبعدها أن البروتوكول بعد عملية التحسين يحقق نسبة تسليم الرزم (PDR) أعلى بمقدار 1.1% من البروتوكول قبل التحسين ، ونلاحظ أن مع زيادة سرعة العقد المتحركة في الشبكة يحدث المزيد من الانقطاعات في مسارات التوجيه نتيجة التغير المتكرر لمواقع العُقد وهذا ينتج عنه فقدان بعض الرزم المُرسَلة مما يؤدي إلى انخفاض نسبة تسليم الرزم (PDR) في الشبكة ، ويوضح الشكل (6 – 5) هذه المقارنة :



الشكل (6 – 5) مقارنة نسبة (PDR) بين البروتوكول قبل عملية التحسين وبعدها عند زيادة سرعة العقد المتحركة

وأخيراً نجد من الجدول (6–4) عند مقارنة نسبة الفائض (Overhead) بين البروتوكول قبل عملية التحسين وبعدها أن البروتوكول بعد عملية التحسين يزيد نسبة الفائض (Overhead) في الشبكة بمقدار 39.96% ، ونلاحظ أن بزيادة سرعة العقد تزداد نسبة الفائض في الشبكة وذلك لأن التبدل المستمر لمواضع العقد في المسارات ينتج عنه عمليات إرسال إضافية لرسائل التحكم من أجل اكتشاف المسارات الجديدة واختيار المسار الأفضل بعد كل تغير في مواضع العُقد ، ويوضح الشكل (6 – 6) نتائج هذه المقارنة :



الشكل (6 – 6) مقارنة نسبة Overhead بين البروتوكول قبل وبعد عملية التحسين عند زيادة سرعة العقد المتحركة

7- الاستنتاجات والتوصيات

7-1- السيناريو الأول (تغير عدد العقد المتحركة في الشبكة) :

من خلال تطبيق السيناريو الأول توصلنا إلى الاستنتاجات والتوصيات التالية :

- عند مقارنة التأخير الزمني (Delay) بين البروتوكول قبل عملية التحسين وبعدها نجد أنه مع زيادة عدد العقد المتحركة في الشبكة ينخفض التأخير الزمني، وإن البروتوكول الذي تم تحسينه في هذا البحث يحقق نسبة تأخير زمني أقل بمقدار 2.32% من البروتوكول قبل عملية التحسين.
- وعند مقارنة نسبة تسليم الرزم (PDR) بين البروتوكول قبل عملية التحسين وبعدها نجد أن البروتوكول الذي تم تحسينه يحقق نسبة PDR أعلى بمقدار 0.38% من البروتوكول قبل عملية التحسين.

- وأخيراً عند مقارنة نسبة الفائض (Overhead) بين البروتوكول قبل عملية التحسين وبعدها نجد أن البروتوكول الذي تم تحسينه يزيد نسبة الفائض في الشبكة بمقدار 32.53% .

7-2- السيناريو الثاني (تغير سرعة العقد المتحركة):

من خلال تطبيق السيناريو الثاني توصلنا إلى الاستنتاجات والتوصيات التالية :

- عند مقارنة التأخير الزمني (Delay) بين البروتوكول قبل عملية التحسين وبعدها نجد أنه مع زيادة سرعة العقد المتحركة في الشبكة يزداد التأخير الزمني، وإن البروتوكول الذي تم تحسينه في هذا البحث يحقق نسبة تأخير زمني أقل بمقدار 8.66% من البروتوكول قبل عملية التحسين.
- وعند مقارنة نسبة تسليم الرزم (PDR) بين البروتوكول قبل عملية التحسين وبعدها نجد أنه مع زيادة سرعة العقد المتحركة في الشبكة تنخفض مسبة تسليم الرزم (PDR) في الشبكة ، وإن البروتوكول الذي تم تحسينه يحقق نسبة (PDR) أعلى بمقدار 1.1% من البروتوكول قبل عملية التحسين.
- وأخيراً عند مقارنة نسبة الفائض (Overhead) بين البروتوكول قبل عملية التحسين وبعدها نجد أن البروتوكول الذي تم تحسينه يزيد نسبة الفائض في الشبكة بمقدار 39.96%.

8-الآفاق المستقبلية

- دراسة تأثير طاقة العقد على جودة المسار، فمن الممكن انهيار المسار نتيجة نفاذ طاقة أحد العقد الموجودة ضمنه حيث يعتبر استهلاك الطاقة أحد العوامل المهمة في البيئات التي يصعب الوصل إليها.
- دراسة احتمال وجود ضجيج على المسار، فمن الممكن أن يكون مسار ما هو الأفضل من حيث طول المسار ولكن ممكن أن يتعرض المسار إلى ضجيج يؤدي إلى فقدان الإشارة .
- بالإضافة إلى دراسة طول المسار بالاعتماد على عدد القفزات، من الممكن دراسة أثر المسافة بين العقد على أفضلية المسار.

9- المراجع

[1] Macker, J.P. and Corson, M.S., 2004. Mobile ad hoc networks (MANETs): Routing technology for dynamic wireless networking. *Mobile Ad hoc networking*, *9*, p 255-273.

[2] Kaur, H., Sahni, V. and Bala, M., 2013. A survey of reactive, proactive and hybrid routing protocols in MANET: A review. *network*, *4*(3), p 498-500.

[3] Dhenakaran, S.S. and Parvathavarthini, A., 2013. An overview of routing protocols in mobile ad-hoc network. *International Journal of Advanced Research in Computer Science and Software Engineering*, *3*(2).

[4] Kaur,H. ; Sahni,V. ; Bala,M 2013. *A Survey of Reactive, Proactive and Hybrid Routing Protocols in MANET: A Review,* International Journal of Computer Science and Information Technologies. Vol 4 ,p 498-500.

[5] Patel,D. N. ; Patel,S. B. ; Kothadiya,H. R. ; Jethwa, P. D. ; Jhaveri,R. H. 2015 , *A survey of reactive routing protocols in MANET.* International Conference on Information Communication and Embedded Systems India. Vol 4 , p 110-126.

[6] Yadav, S. and Lal, P., 2019, March. Multi Point Relay Selection in OLSR Routing Protocol using Particle Swarm Optimization. In *Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE)*.

[7] Gutjahr, W.J., 2003, September. A converging ACO algorithm for stochastic combinatorial optimization. In *International*

Symposium on Stochastic Algorithms (pp. 10-25). Springer, Berlin, Heidelberg.

[8] Godbole, V. (2013). Performance analysis of bio-inspired routing protocols based on random waypoint mobility model. Defence S & T Technical Bulletin, Science & Research Technology Institute for Defence (STRIDE), Vol. 5, No. 2, , pp. 114-134.

[9] Gupta, S.G., Ghonge, M.M., Thakare, P.D. and Jawandhiya, P.M., 2013. Open-source network simulation tools: An overview. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, *2*(4), pp.1629-1635.

[10] Gautam, G. and Sen, B., 2015. Design and simulation of wireless sensor network in NS2. *International Journal of Computer Applications*, *113*(16).

[11] Sirisala,S. ; Ramakrishna,S. *Survey: Enhanced Trust Management for Improving QoS in MANETs.* First International Conference on Artificial Intelligence and Cognitive Computing India. Vol 2,2018, p 255–263.

كشف نوبات الصرع من إشارات الدماغ EEG باستخدام LPC وLS-SVM

د. م. ألفت جولحة*

م. زينب دعدع**

ملخص

وفقاً لمنظمة الصحة العالمية WHO، مرض الصرع هو أحد الأمراض الأكثر شيوعاً التي تصيب الجهاز العصبي المركزي، ويتميز بحدوث نوبة الصرع بشكل مفاجئ. وإنّ القدرة على كشف النوبة بشكل سريع ودقيق آلياً سيدفع للحصول على لمساعدة الطبية الفورية ويتفادى الإصابات الناتجة عنها. يستخدم في هذا البحث الترميز التنبؤي الخطي LPC (Linear Predictive Coding) لاستخلاص سمات نوبات الصرع من إشارات الدماغ EEG (Electroencephalogram). ثم يجرى تمرير تلك السمات إلى مصنف LS-SVM (Least Square-Support Vector Machine) لكشف وجود نوبة الصرع من عدم وجودها باستخدام برنامج MATLAB. وأظهرت النتائج تحسناً ملحوظاً في دقة كشف نوبات الصرع بنسبة 98.66% بينما كانت أعلى نسبة كشف في الدراسات السابقة هي 97.6%.

الكلمات المفتاحية: كشف نوبة الصرع، الترميز التنبؤي الخطي LPC، LS-SVM.

*مدرس، قسم هندسة الحاسبات والتحكم الآلي، كلية الهندسة الميكانيكية والكهربائية، جامعة تشرين، اللاذقية، سورية.

**طالبة دراسات عليا(ماجستير)، قسم هندسة الحاسبات والتحكم الآلي، كلية الهندسة الميكانيكية والكهربائية، جامعة تشرين، اللاذقية، سورية.

Epilepsy Seizures Detection from EEG Signals Using LPC and LS-SVM

Dr. Oulfat Jolaha*
Eng. Zeinab Dadaa**

Abstract

According to World Health Organization (WHO), epilepsy is one of the most common primary diseases of the central nervous system, which characterizes by epileptic seizures. Thus, the ability to automatically detecting epileptic seizure leading to fast medical assistance. In this research, LPC (Linear Predictive Coding) was used for features extraction of epileptic seizures in EEG (Electroencephalography) signals. Then these features were classified using Least Square-Support Vector Machine (LS-SVM) classifier to detect epilepsy seizure. MATLAB was used for modeling the proposed system, results showed a noticeable improvement in detection epilepsy seizures compared with previous studies.

Keywords**:** Epileptic seizures Detection, LPC, LS- SVM Classifier.

*PhD Doctor, Department of Computer and Automatic Control Engineering, Faculty of Mechanical and Electrical Engineering, Tishreen University, Lattakia, Syria.

**Postgraduate Student, Department of Computer and Automatic Control Engineering, Faculty of Mechanical and Electrical Engineering, Tishreen University, Lattakia, Syria.

1- مقدمة

تعتبر إشارة الدماغ Electroencephalogram (EEG) مهمة جداً لتشخيص مرض الصرع، حيث تحتوي تسجيلات EEG لمريض الصرع على كمية كبيرة من البيانات. لذلك يتطلب كشف البؤر الصرعية تحليلاً كاملاً لإشارات الدماغ EEG وغالباً ما تنجز من قبل الخبراء [1]. بذلت جهود عديدة ونشرت أبحاث كثيرة للكشف الآلي لنوبات الصرع التي تختلف من مريض لآخر ومن نوبة لأخرى عن طريق تسجيلات إشارات الدماغ EEG وECOG. ومن الدراسات التي استخدمت طرق تعلم الآلة الدراسة [2] التي صنفت إشارة ECOG إلى إشارات تدل على وجود صرع من عدم وجوده باستخدام آلية شعاع الدعم SVM (Support Vector Machine) وطرق تحليل المكونات PCA (Principal component analysis) وICA(Independent Component Analysis)، وجد أنها تعطي حساسية 98% ونوعية 80%، ولكنها تستغرق زمنَ تشغيلٍ كبيرٍ لتحليل المكونات. تم تحليل إشارات الدماغ EEG وتصنيفها لاكتشاف مستويات مخاطر الصرع باستخدام مصنفين: الأول مصنف Fuzzy يعمل بالاعتماد على الميزات المستخلصة من إشارات دماغ المريض مثل الطاقة والتباين والحدّة في موجات الدماغ، والمصنف الثاني مصنف آلية شعاع الدعم SVM والإنتروبيا النسبية بحدودها الدنيا وذلك لتحقيق الأمثلية وتحسين التصنيف، وتم استخدام بيانات 10 مرضى وكانت الحساسية 97.07% والنوعية 97% [3]. استخدم التعلم العميق لكشف وتصنيف نوبات الصرع باستخدام CNN (Convolution Neural Network) فكانت الدقة 88.6% والنوعية 90% والحساسية 95% [4]. تم كشف تغير معدل نبضات القلب المتعلقة بنوبات الصرع باستخدام مراقب قلبي صالح للارتداء يعمل لاسلكياً وتم الكشف باستخدام WiSARD Neural Network

وDWT (Discrete Wavelet Transform)، فكانت الدقة جيدة ولكن الأجهزة اللاسلكية تستهلك طاقة كبيرة [5] . وكذلك تم استخدام كلٍ من DT (Decision Tree) وSVM و DWT وCWT (Continuous Wavelet Transform) وRF (Random Forest) و ANN (Artificial Neural Network) لاستخلاص السمات وكشف نوبات الصرع فكانت الدقة لكل من خوارزمية SVM وKNN وDT وANN هي 97.6% و97% و97.6% و97.4%، على التوالي [6]. ووجد أن استخدام DWT متعدد المستويات وبحزم مختلفة وبتطبيق التوابع الإحصائية الآتية: variance وmean وSkewness و MAV (Mean Absolut Value) وSD (Standard Deviation) وShannon entropy يستهلك زمناً حسابياً كبيراً [6]. في حين أنه عند استخدام CNN وCWT في كشف نوبة الصرع من إشارات الدماغ EEG كانت دقة التصنيف 93.5% [7]. واستخدم FFT (Fast Fourier Transform) و STFT (short-time Fourier Transform) وWT (Wavelet Transforms) مع CWT وCNN لكشف وتصنيف نوبة الصرع من إشارات الدماغ EEG فكانت الدقة 93.60% [8].

يلاحظ في الدراسات المذكورة أن أعلى دقة كشف تم التوصل إليها هي 97.6% [6]، لذا استخدم في هذا البحث LPC ومصنف LS-SVM من أجل زيادة دقة الكشف بالنسبة للدراسات المرجعية السابقة.

2- هدف البحث وأهميته

بناء نظام لتحسين دقة كشف نوبات الصرع باستخدام الترميز التنبؤي الخطي LPC ومصنف LS-SVM. يمكّن هذا النظام من حصول المصاب على المساعدة الطبية الفورية مما يقيه من الإصابات الناتجة عن النوبة. كما أن تحليل EEG المنجز بصرياً من قبل الأطباء يستهلك وقتاً، لذلك فإن تقنيات معالجة إشارات الدماغ EEG تساعد في تسريع هذه العملية وتسمح للمستخدمين الطبيين بتمييز حالات الصرع بدقة.

3- مواد وطرق البحث

اعتمدت قاعدة بيانات من جامعة Bonn الألمانية [12]، وهي تتألف من 5 مجموعات لإشارات تخطيط دماغ، كل مجموعة منها تتألف من 100 مقطع، مدة كل مقطع 23.6 ثانية أخذت من الأشخاص باستخدام 32 قطب كالآتي [9]:

- المجموعتان F,N: أخذت من أشخاص بحالة Inter Ictal.
- المجموعتان Z,O: أخذت من أشخاص أصحاء.
- المجموعة S: أخذت من أشخاص بحالة Ictal.

استخدم برنامج MATLAB R2014a وحاسب بمعالج N2810 @2.00GHz.

مرض الصرع هو اختلال عصبي داخلي ينتج عن اضطرابات الخلايا الكهربائية في خلايا المخ والخلل القائم في العملية الكهربائية الدماغية، والعَرَض الأساسي الجامع لكل أشكال الصرع هو فقدان الوعي بالإضافة إلى حدوث تشنجات ونوبات أحياناً [1]. وتقسم النوبة الصرعية إلى المراحل الآتية [1] و[9]:

1. Pre-Ictal: تحدث تماماً قبل 30 إلى 60 دقيقة من بداية نوبة الصرع.
2. Ictal: تعرف كنوبة صرع تستمر من 1 إلى 3 دقائق يضعف خلالها نشاط الدماغ.
3. Inter-Ictal: تحدث بين نوبتي صرع متتاليتين ويمكن أن تميز نشاط دماغي شاذ أو طبيعي.
4. Post-Ictal: هي الحالة التي يتعافى فيها الدماغ من النوبة الصرعية وتتراوح مدتها بين 30 إلى 60 دقيقة بعد النوبة الحقيقية.

إشارة تخطيط الدماغ EEG هي تسجيل للنشاط الكهربائي التلقائي للدماغ خلال فترة من الزمن. تتركز الاستعمالات التشخيصية بشكل عام على المحتوى الطيفي لتخطيط أمواج الدماغ. ويستخدم تخطيط الدماغ بشكل كبير لتشخيص مرض الصرع الذي يسبب أنماطاً

غير طبيعية في قراءات التخطيط الدماغي. إن وجود نشاطات صرع في إشارات EEG يؤكد تشخيص مرض الصرع عند المريض، وهذه النشاطات تشبه إطلاق شرارة spike أو موجات حادة sharp wave، تتراوح مدة الشرارة بين 20−70 ميلي ثانية وتتبعها موجة حادة لمدة 70−200 ميلي ثانية [1] و[9].

3-1 نظام كشف نوبات الصرع المقترح

أول مرحلة في نظام كشف نوبات الصرع المقترح هي إزالة الضجيج باستخدام مرشح تمرير منخفض يعمل بتردد قطع Cutoff=Fs\8= 21.7 HZ حيث Fs= 173.6 HZ وذلك لأن الترددات التي تعطي معلومات مهمة عن نوبات الصرع تقع في المجال الترددي لموجات إلفا (α) ودلتا(δ) الدماغية التي تتراوح تردداتها بين [0-4 HZ] لموجات (α) و[8-16 HZ] لموجات ودلتا(δ).

وفي المرحلة الثانية يتم تحويل إشارات الدماغ EEG الناتجة إلى المجال الترددي باستخدام تحويل فورييه السريع FFT، وهو تحويل يحسب قيمة تحويل فورييه المتقطع DFT بسرعة، وتعود سرعته لعدم حسابه للأجزاء التي مجموعها يساوي صفر في تحويل فورييه المتقطع [10]. وتبنى فكرة هذا التحويل على تجزئة التتابع $s(n)$ إلى عدد من التتابعات الفرعية الأصغر طولاً حيث n متغير في المجال الزمني. ولإيضاح مبدأ التقسيم في الزمن نعتبر الحالة الخاصة عندما يكون $N = 2^M$ حيث M عدد صحيح موجب أكبر من الواحد وبما أن N عدد زوجي يمكن تجزئة التتابع $s(n)$ إلى تتابعين فرعيين متساويين كل منهما بطول $N/2$، التتابع الأول $g(n)$ من العينات ذات الترتيب الزوجي$0, 2, 4, \dots, N-2$ والتتابع الثاني $h(n)$ من العينات ذات التتابع الفردي $1, 3, 5, \dots, N-1$ وفق المعادلة الآتية [10]:

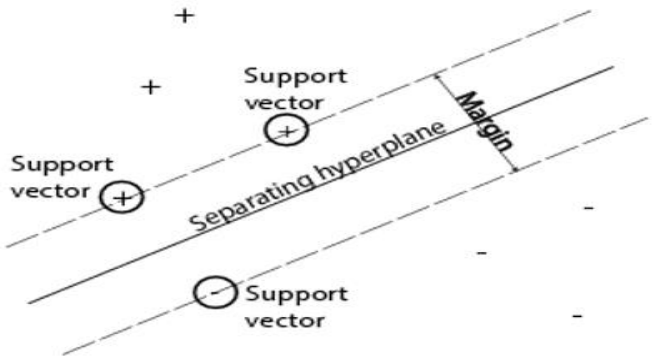
$$s(k) = \sum_{r=0}^{\frac{N}{2}-1} s(2r)\, W_{N/2}^{rk} + W_N^K \sum_{r=1}^{\frac{N}{2}-1} s(2r+1)\, W_{N/2}^{rk} \qquad (1)$$

حيث $W_{N/2} = e^{-j\frac{2\pi}{N/2}}$ و $r = 0.1, 2, 3, \dots, \frac{N}{2} - 1$

وبعد الحصول على الإشارة في المجال الترددي يتم استخلاص السمات باستخدام الترميز التنبؤي الخطي LPC، الذي يعتمد على فكرة أنّ العينة الحالية يمكن أن تكون بشكل تقريبي مجموعة خطية من العينات السابقة توصف رياضياً بالعلاقة [11]:

$$s(n) \approx \mathrm{a}_1 s(n-1) + \mathrm{a}_2 s(n-2) + \cdots + \mathrm{a}_p s(n-2) \qquad (2)$$

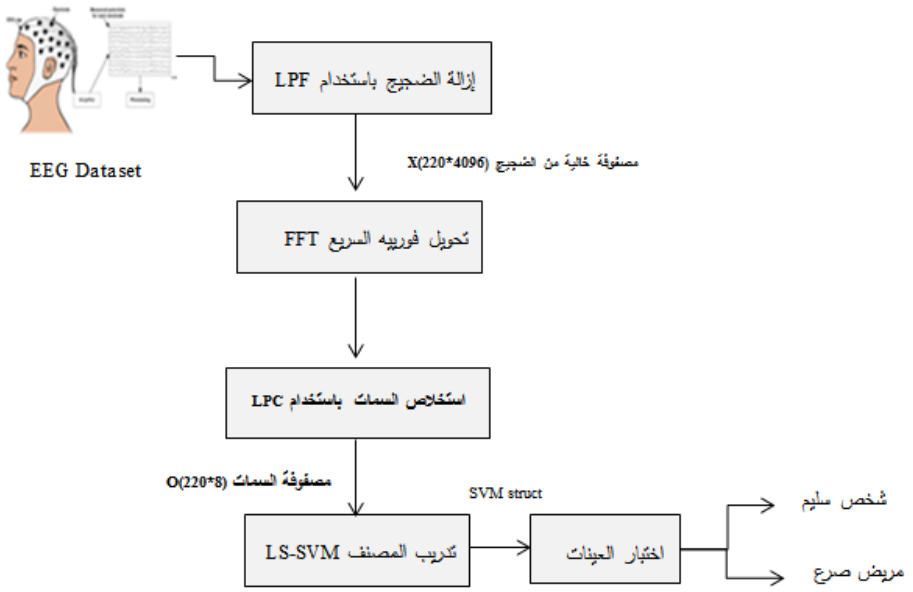
حيث تمثل n طول السلسلة وp يمكن أن تكون عدداً ما من 1 إلى $n-1$ والترميز التنبؤي الخطي يحسب مجموعة المعاملات $\{\mathrm{a_k}\}$ التي تصف التشكيلات الموجية المميزة Formants، وهي تمثل الترددات عند القمم الاهتزازية resonant peaks وتدعى ترددات الموجة المميزة. يتم تخمين مواقع هذه الترددات المميزة في إشارة EEG عن طريق حساب معاملات التنبؤ الخطي $\{\mathrm{a_k}\}$ على طول الإشارة وإيجاد القمم في الطيف الناتج عن مرشح النموذج التنبؤي الخطي. ومن ثم تبدأ مرحلة تدريب المصنف LS-SVM الذي يعمل على تصغير مجموع مربعات الأخطاء للتابع الهدف. يقوم مصنف SVM في حالة التصنيف الثنائي بإيجاد أفضل مستوي يفصل جميع نقاط الصنف الأول عن جميع نقاط الصنف الثاني، وأفضل مستوي حدي هو المستوي الذي يملك أكبر هامش بين الصنفين، ويعني الهامش العرض الأعظمي للكتلة الموازية للمستوي التي ليس لها نقاط بيانات داخلية وموجهات الدعم هي نقاط البيانات الأقرب إلى مستوي الفصل وهذه النقاط على حد الكتلة الموازية لمستوي الفصل، وهذا مبين في الشكل (1) [3]. تستخدم ثلاث طرق في مصنف SVM هي البرمجة الخطية من الدرجة الثانية QP، والتحسين التسلسلي الأدنى SMO وهي خوارزمية لحل مشكلة البرمجة التربيعية التي تنشأ أثناء تدريب المتجهات الداعمة وهي الطريقة الافتراضية للمصنف SVM، ولقد جرى في هذا البحث استخدام طريقة المربعات الصغرى LS-SVM التي يتم فيها تحليل البيانات والتعرف على الأنماط.



الشكل (1) مصنف SVM [3].

3-2 مخطط نظام كشف نوبات الصرع المقترح

يبين المخطط في الشكل (2) خطوات كشف نوبات الصرع في النظام المقترح حيث يتم إدخال بيانات إشارات EEG وبعدها يتم إزالة الضجيج منها باستخدام المرشح filterfft وهو مرشح تمرير منخفض LPF الذي ينتج مصفوفة خالية من الضجيج X(220*4096)، ومن ثمّ يتم تحويل الإشارات إلى المجال الترددي باستخدام تحويل فورييه السريع FFT، ليتم استخلاص سمات المصفوفة X باستخدام الترميز التنبؤي الخطي LPC وتنتج مصفوفة السمات.التي تمثل معاملات التنبؤ الخطي O(220*8) وتستخدم في تدريب المصنف LS-SVM. وبعدها يتم تدريب المصنف واختباره على مجموعة من العينات ليكون خرج الاختبار إما شخص مصاب بالصرع أو شخص سليم. من المهم في إشارة EEG أن تكون الإشارة موصوفة بصيغة رياضية سهلة وبسيطة مع الاحتفاظ بكل خصائص الإشارة لذلك تم استخدام LPC في هذا البحث.

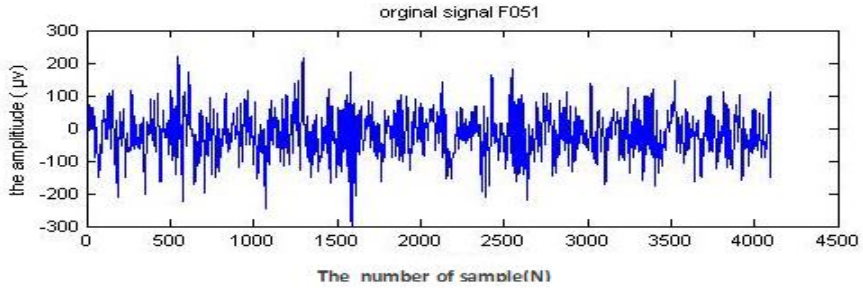


الشكل (2) مخطط نظام كشف نوبات الصرع المقترح.

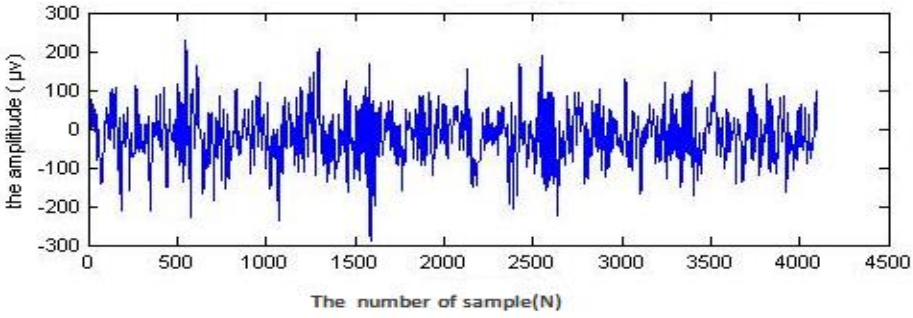
4- النتائج ومناقشتها

جرى إدخال 220 إشارة EEG مأخوذة من قاعدة البيانات لمجموعة أشخاص أصحاء وأشخاص في حالة نوبة الصرع Inter-Ictal وIctal على نظام كشف نوبات الصرع المقترح، حيث تم إزالة الضجيج من إشارات EEG المدخلة باستخدام مرشح تمرير منخفض LPF، ومن ثمّ تحويل الإشارات للمجال الترددي باستخدام تحويل فورييه السريع FFT وبعد ذلك تم استخلاص السمات باستخدام الترميز التنبؤي الخطي LPC بأكثر من درجة. استخدم برنامج MATLAB في نمذجة نظام كشف نوبات الصرع المقترح، وتم عرض 3 إشارات من إشارات EEG الأصلية فقط كأمثلة مع نتيجة إزالة الضجيج منها وعرض نتائج الترميز التنبؤي الخطي LPC باستخدام درجات مختلفة وذلك في الأشكال من الشكل (3) إلى الشكل (17)، ولوحظ أن الدرجة 8 هي الأنسب فالدرجات الأكبر من هذه الدرجة تعطي تغيرات في الإشارة قليلة جداً ولا تفيد في التصنيف.

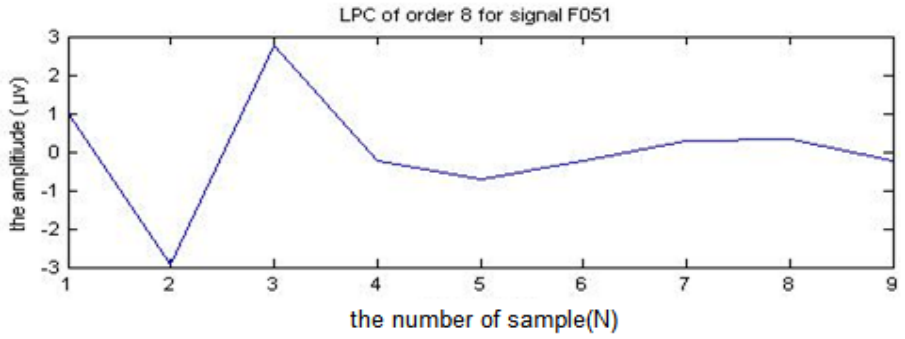
يبين الشكل (3) إشارة EEG الأصلية للإشارة F051 وهي لشخص مريض صرع بحالة النوبة INTER ICTAL. نلاحظ وجود موجات عابرة بشكل شرارات مفردة أو متتالية وموجات حادة أو موجات حادة تتبعها شرارة ولكنّها بشكل عابر ولا تظهر على طول التسجيل الكهربائي لإشارة EEG. بينما يبين الشكل (4) الإشارة F051 بعد إزالة الضجيج منها باستخدام مرشح تمرير منخفض LPF بتردد قطع 21.7 HZ. والأشكال (5) و(6) و (7) تبين معاملات التنبؤ الخطي المستخلصة من الإشارة F051 بدرجات تنبؤ مختلفة، التي تمثل قيم تخمينية لمواقع الترددات المميزة عند القمم الاهتزازية في طيف الإشارة.



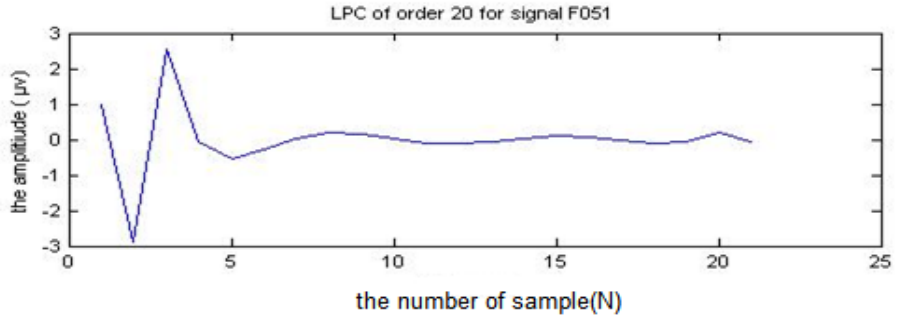
الشكل (3) إشارة EEG الأصلية للإشارة F051.



الشكل (4) إشارة F051 خالية من الضجيج.

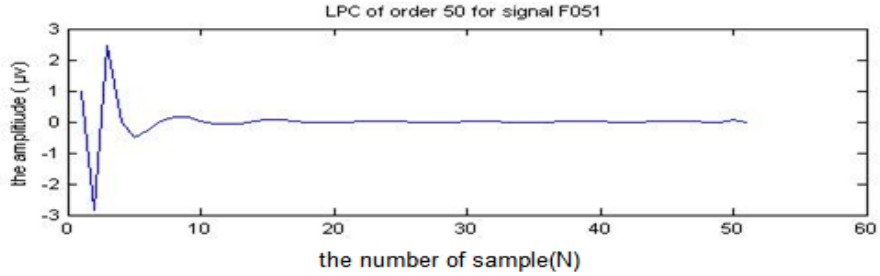


الشكل (5) الترميز التنبؤي الخطي للإشارة F051 من الدرجة 8.

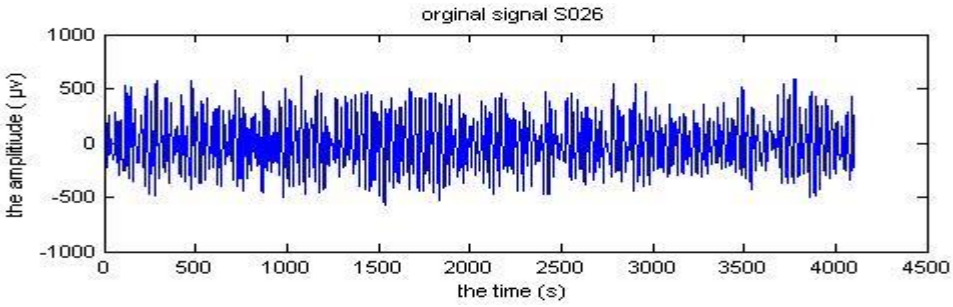


الشكل (6) الترميز التنبؤي الخطي للإشارة F051 من الدرجة 20.

يبين الشكل (8) إشارة EEG الأصلية للإشارة S026 وهي لشخص مريض صرع بحالة النوبة ICTAL. نلاحظ وجود إطلاقات مستمرة من الموجات غير المنتظمة والمطالات والترددات المتغيرة ووجود شرارات تتبعها موجات حادة معقدة ووجود فرط تزامن إيقاعي أو خمول في كهربائية الدماغ أطول من معدلاته خلال حالات InterIctal. **والشكل (9)** يبين الإشارة S026 بعد إزالة الضجيج منها باستخدام مرشح تمرير منخفض LPF بتردد قطع 21.7 HZ. الأشكال (10) و(11) و (12) تبين معاملات التنبؤ الخطي المستخلصة من الإشارة F051 بدرجات تنبؤ مختلفة، التي تمثل قيم تخمينية لمواقع الترددات المميزة عند القمم الاهتزازية في طيف الإشارة.

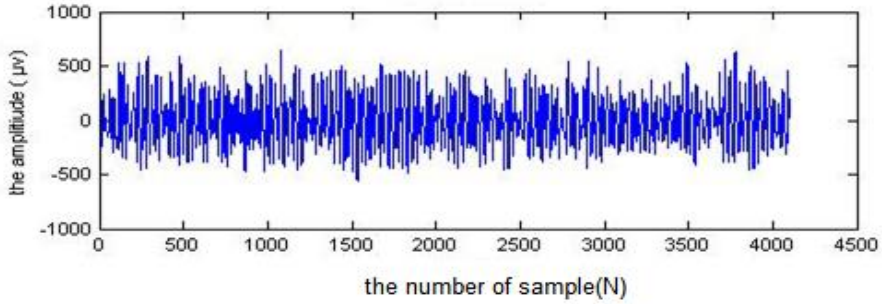


الشكل (7) الترميز التنبؤي الخطي للإشارة F051 من الدرجة 50.

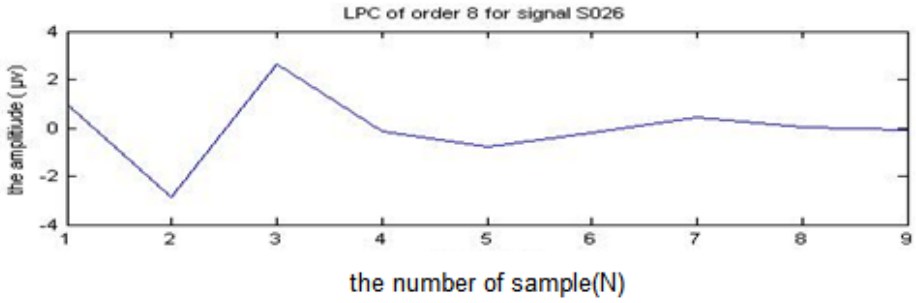


الشكل (8) إشارة EEG الأصلية للإشارة S026.

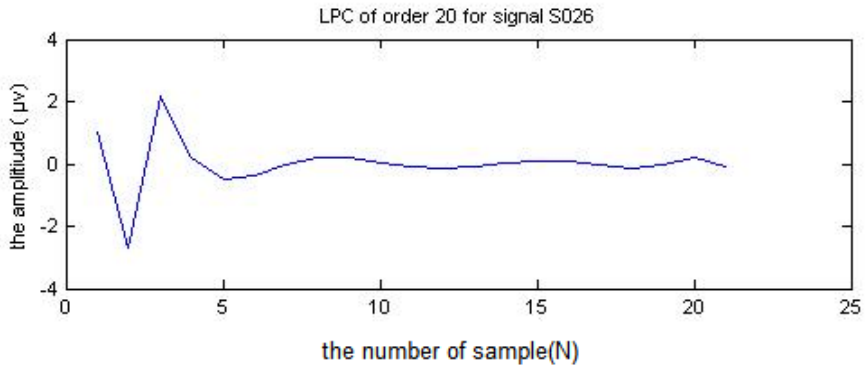
يبين الشكل (13) إشارة EEG الأصلية للإشارة F051 وهي لشخص سليم ونلاحظ أنّها إشارة دماغ طبيعية لا يوجد فيه شذوذ أو موجات غير طبيعية. والشكل (14) يبين الإشارة O079 بعد إزالة الضجيج منها باستخدام مرشح تمرير منخفض LPF بتردد قطع 21.7HZ.



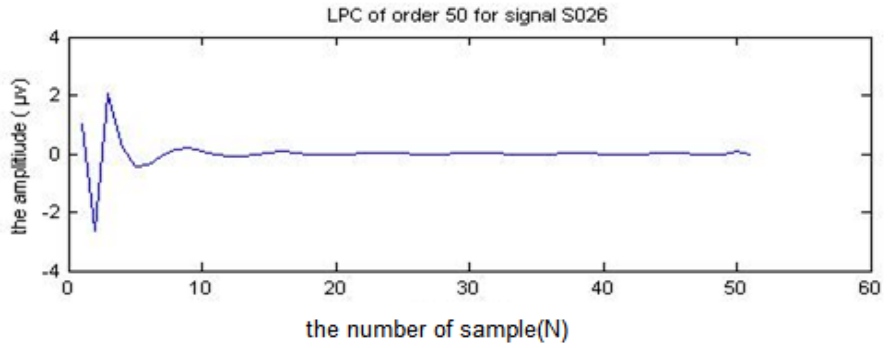
الشكل (9) إشارة S026 خالية من الضجيج.



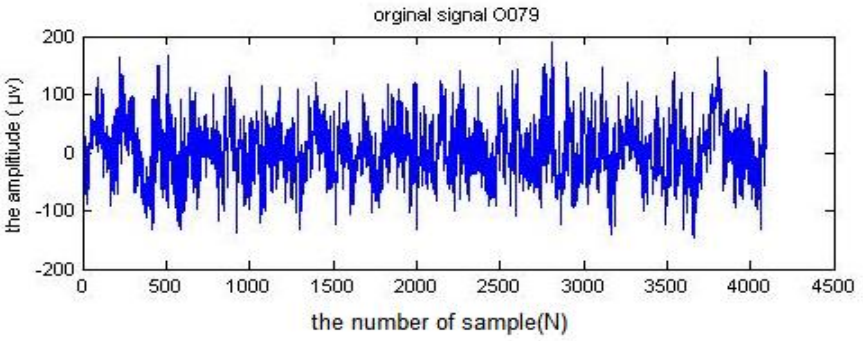
الشكل (10) الترميز التنبؤي الخطي للإشارة S026 من الدرجة 8.



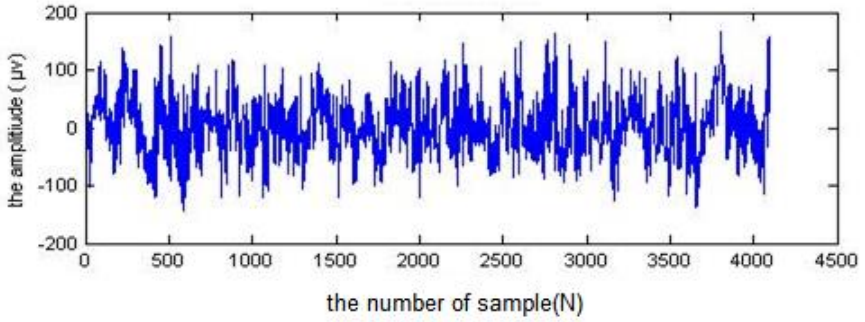
الشكل (11) الترميز التنبؤي الخطي للإشارة S026 من الدرجة 20.



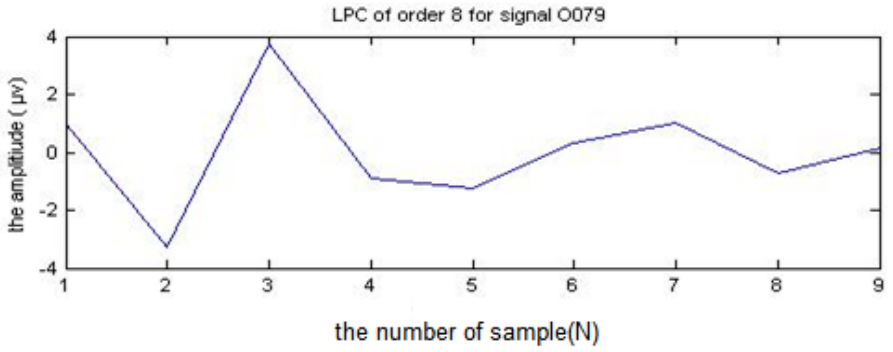
الشكل (12) الترميز التنبؤي الخطي للإشارة S026 من الدرجة 50.



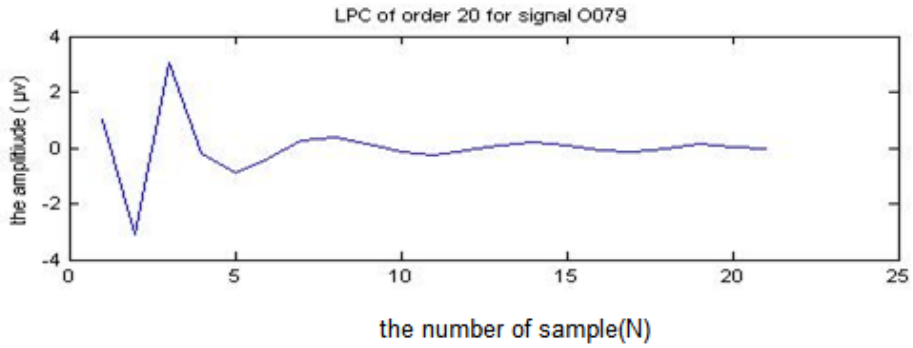
الشكل (13) إشارة EEG الأصلية للإشارة O079.



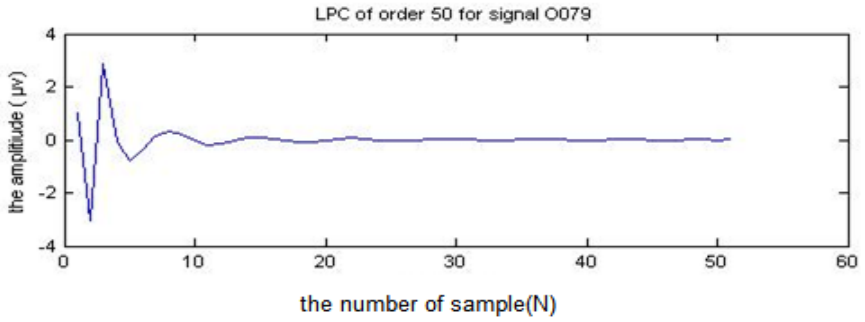
الشكل (14) إزالة الضجيج للإشارة O079.



الشكل (15) الترميز التنبؤي الخطي للإشارة O079 من الدرجة 8.



الشكل (16) الترميز التنبؤي الخطي للإشارة O079 من الدرجة 20.



الشكل (17) الترميز التنبؤي الخطي للإشارة O079 من الدرجة 50.

بينما تبين الأشكال (15) و(16) و (17) معاملات التنبؤ الخطي المستخلصة من الإشارة O079 بدرجات تنبؤ مختلفة، التي تمثل قيم تخمينية لمواقع الترددات المميزة عند القمم الاهتزازية في طيف الإشارة.

تم تدريب المصنف LS-SVM باستخدام 220 عينة تدريب ومن ثم اختباره على 250 عينة من قاعدة البيانات. يبين الجدول (1) نتائج التصنيف لبعض عينات EEG وتوافقه مع التصنيف الحقيقي للإشارة الذي يمثل رأي الخبراء حسب قاعدة البيانات المستخدمة. بينما الجدول (2) يعطي نتائج اختبار المصنف على مجموعات جزئية من قاعدة

البيانات. ولدراسة أداء أي مصنف في مجال التحليل التنبؤي تعد مصفوفة الارتباك (confuse matrix) من أفضل المقاييس. يبين الشكل (18) عناصر مصفوفة الارتباك وهي [13]:

- True Positive (TP) نحصل عليه عندما يكون تصنيف المصنف ايجابي والتصنيف الحقيقي ايجابي بمعنى أن المصنف أصاب 'hit'.
- True Negative (TN) نحصل عليه عندما يكون تصنيف المصنف للعينة سلبي والتصنيف الحقيقي سلبي أي بمعنى الرفض الصحيح correct rejction .
- False Positive (FP) نحصل عليها عندما يكون تصنيف المصنف للعينة ايجابي والتصنيف الحقيقي سلبي أي خطأ نوع I .
- False Negative (FN) نحصل عليه عندما يكون تصنيف المصنف للعينة سالب والتصنيف الحقيقي موجب أي خطأ من النوع II.

		Ground Truth	
		Positive	Negative
Prediction	Positive	True Positives	False Positives
	Negative	False Negatives	True Negatives

الشكل (18) مصفوفة الارتباك [13].

تعرف حساسية $sensitivity$ المصنف بأنها النسبة المئوية للأشخاص المصابين بالمرض وتم تشخيص إصابتهم بالمرض بشكل صحيح من قبل المصنف وتحسب بتطبيق العلاقة [13]:

$$sensitivity = \frac{TP}{TP+FN} * 100\% \qquad (3)$$

والنوعية $specifity$ هي النسبة المئوية للأشخاص الأصحاء الذين تم تشخيصهم بأنهم غير مصابين بالمرض من قبل المصنف وتحسب بتطبيق العلاقة [13]:

$$specifity = \frac{TN}{TN+FP} * 100\% \qquad (4)$$

أما الدقة $precision$ فتعبر عن نسبة النواحي الإيجابية المميزة بشكل صحيح وتحسب بتطبيق العلاقة [13]:

$$precision = \frac{TP}{positive\ output} \qquad (5)$$

يبين الجدول (1) نتائج التصنيف لبعض عينات EEG المستخدمة في قاعدة البيانات. بينما يقدم الجدول (2) تقييماً لأداء نظام كشف نوبات الصرع المقترح، فمثلاً من أجل المجموعة F+N+O يعطي نظام الكشف المقترح دقة 99% وحساسية 100% ونوعية 98%، وعند اختباره على إشارات من جميع مجموعات قاعدة البيانات F+N+S+Z+O يعطي دقة 98.66% وحساسية 98.66% ونوعية 98%. ومن خلال مقارنة هذه النتائج مع الدراسات السابقة كما في الجدول (3) نلاحظ تحسّناً في دقة التصنيف والحساسية والنوعية للمصنف المقترح وذلك لأن استخدام LPC في تحليل السلاسل الزمنية دقيق رياضياً وبسيط في الحساب والتطبيق ويحتفظ بكل خصائص الإشارة.

حيث تم حساب بارامترات الأداء باستخدام معادلات نفذت باستخدام برنامج MATLAB وفق العلاقات (3) و(4) و(5) وذلك بعد الحصول على عناصر مصفوفة الارتباك.

نلاحظ من الجدول (3) أن النظام المقترح تفوق على النظام في الدراسة [2] من ناحية النوعية والحساسية وعلى الأنظمة في الدراسات [3] و[4] و[6] و[8] من ناحية الدقة. ورغم أن الحساسية في النظام [4] هي 100% إلّا أن نظام كشف نوبات الصرع المقترح قد حقق نوعية ودقة أكبر وذلك لأن المجموعة الأفضل من معاملات التنبؤ الخطي تقلل مربع متوسط الخطأ التنبؤي في إشارات EEG وطريقة المربعات الصغرى LS المستخدمة في مصنف SVM تعمل على اختيار الخط المستقيم الأفضل الذي يحقق أن مربعات أبعاد النقاط عن هذا الخط أقل ما يمكن.

الجدول(1) نتائج التصنيف لبعض عينات EEG المستخدمة في قاعدة البيانات.

رقم العينة	رأي الخبراء	نتيجة المصنف	التوافق
F051	مريض صرع	مريض صرع	√
F052	مريض صرع	مريض صرع	√
F053	مريض صرع	مريض صرع	√
F054	مريض صرع	مريض صرع	√
F055	مريض صرع	مريض صرع	√
F060	مريض صرع	مريض صرع	√
N026	مريض صرع	مريض صرع	√
S056	مريض صرع	مريض صرع	√
S058	مريض صرع	مريض صرع	√
S059	مريض صرع	مريض صرع	√
S060	مريض صرع	شخص سليم	×
Z039	شخص سليم	مريض صرع	×
Z040	شخص سليم	شخص سليم	√
Z041	شخص سليم	شخص سليم	√
O060	شخص سليم	شخص سليم	√

الجدول (2) تقييم الأداء لنظام كشف نوبات الصرع المقترح.

مجموعة الاختبار	الدقة	الحساسية	النوعية	Confusion Matrix Parameters
F+N+S	98.3%	98.66%	98%	TP=148 FN=2 TN=98 FB=2
F+N+Z	98.66%	100%	96%	Tp=100 FN=0 FB=2 TN=48
F+N+O	99%	100%	98%	TP=100 FN=0 FB=1 TN=49
F +S+Z	98%	98%	96%	TP=98 FN=2 FB=2 TN=48
F+S+Z+O	98.5%	100%	97%	TP=100 TN=97 FB=3 FN=0

F+N+S+Z+O	98.66%	98.66%	98%	TP=148 TN=98 FB=2 FN=2

الجدول (3) مقارنة بارامترات أداء نظام كشف نوبات الصرع المقترح مع الدراسات السابقة.

رقم الدراسة	الدقة	الحساسية	النوعية
[2]	–	98%	80%
[3]	–	97.07%	97%
[8]	90%	100%	83.3%
[6]	SVM: 97.6% KNN: 97% DT: 97.6% ANN: 97.4%	–	–
[4]	88.67%	90%	95%
[7]	93.5%	–	–
[8]	93.6%	–	–
الدراسة الحالية	98.66%	98.66%	98%

5- الاستنتاجات

جرى استخلاص سمات إشارات EEG لكشف وجود نوبة صرع ووجد أنه باستخدام الترميز التنبؤي الخطي LPC من الدرجة 8 واستخدام مصنف LS-SVM أمكن الحصول على دقة كشف لنوبة الصرع بنسبة 98.66% وحساسية بنسبة 98.66% ونوعية بنسبة 98% على عينات EEG، وهذه النتائج تدل على قوة النظام المقترح في كشف نوبة الصرع بالمقارنة مع الدراسات السابقة.

6- التوصيات والمقترحات

تعديل المصنف السابق لتصنيف جزئي لحالة Pre-Ictal التي تحدث تماما قبل 30 إلى 60 دقيقة من بداية نوبة الصرع وذلك باستخدام قاعدة بيانات حالة النوبة InterIctal التي تحدث بين نوبتين صرعيتين متتاليتين وذلك في حال عدم توفر قاعدة بيانات لحالة

Pre-Ictal مما يسهم بشكل فعال في التنبؤ بحدوث النوبة بدقة وتقديم العلاج اللازم بسرعة.

7- المراجع

[1] ZHOU, J., 2014 - **A Study of Automatic Detection and Classification of EEG Epileptiform Transients**. PhD Thesis, Clemson University, 139p.

[2] BALASUBRAMANIAN, P., 2014 - **Automated Classification of EEG Signals Using Component Analysis and Support Vector Machines**, Master Thesis, Grand Valley State University, 90 p.

[3] GURUMURTHY, S., and TRIPATHY, B. K., 2015 - Classification and Analysis of EEG Brain Signals for Finding Epilepsy Risk Levels Using SVM. **World Applied Sciences Journal**, Vol. 33 (4), 631-639.

[4] ACHARYA, U. R., OH, S. L., HAGIWARA, Y., TAN, J. H., and ADELI, H., 2018 - Deep Convolutional Neural Network for the Automated Detection and Diagnosis Of Seizure Using EEG Signals. **Computers in Biology and Medicine**, Vol. 100, 270–278.

[5] RAO, P.V., AKILAN, S., DHIVAKA, V., and KARTHIKEYAN, D., 2016 - Epilepsy Seizure Detection Using EEG - Curvelet Feature Selection and SVM Classification, **International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE)**, Vol. 5, Issue 3, March.

[6] AL-QEREM, A., KHARBAT, F., NASHWAN, S., ASHRAF, S., and BLAOU, K. 2020 - General Model for Best Feature Extraction of EEG Using Discrete Wavelet Transform Wavelet Family and Differential Evolution, **International Journal of Distributed Sensor Networks**, Vol. 16 (3).

[7] ZHAO, W., et. al., 2020 - A Novel Deep Neural Network for Robust Detection of Seizures Using EEG Signals, **Computational and Mathematical Methods in Medicine**, Hindawi, Vol. 2020.

[8] TÜRK, Ö., and ÖZERDEM, M. S., 2019- Epilepsy Detection by Using Scalogram Based Convolutional Neural Network From EEG Signals, **Brain Sciences, MDPI**, Vol. 9, 115.

[9] PROEKT, A., 2018 - Brief Introduction to Electroencephalograph. **Elsevier Inc., Methods in Enzymology**, Vol. 603, 257-273.

[10] OBERST, U., 2007 - The Fast Fourier Transform. **SIAM Journal and Control and Optimization**, Vol. 46(2), 496-540.

[11] SIWALANKERTO, J., et. al., 2015- Speech Recognition Using Linear Predictive Coding and Artificial Neural Network for Controlling Movement of Mobile Robot, **International Conference on Information and Electronics Engineering (IPCSIT)**, Vol. 6, IACSIT Press, Singapore.

12. http://www.epileptologie-bonn.de. Accessed Nov. 2017.

13. JU´AREZ-GUERRA, E. and ALARCON-AQUINO, V., and Juárez-Guerra, E., Alarcon-Aquino, V., & Gómez-Gil, P., 2011 - Epilepsy Seizure Detection in EEG Signals Using Wavelet Transforms and Neural Networks. **New Trends in Networking, Virtual International**

Conference on Computing, E-Learning, Systems Sciences, and Engineering (CSSE), 261–269.

تطبيق الاختبارات المؤتمتة المستمرّة ضمن خط التكامل والتّسليم المستمر

م. سالي جركس*، د. ناصر أبو صالح**، د. أليدا اسبر***

ملخص

نظراً للمنافسة المتزايدة ومتطلبات السّوق المتغيّرة، تحتاج الشركات إلى المرونة والسّرعة للتكيّف مع التّحديثات الأخيرة وأخذ مكانها في السّوق. لتحقيق ذلك، تعتمد الشركات على تقنيّات وطرائق جديدة مثل منهجيّة أجيل وDevOps كمنهجيّة أساسيّة في الشركة. وعلى الرغم من وجود العديد من الأبحاث التي تستعرض طريقة تحقيق مناهج الأجيل في تطبيق أفكار DevOps إلا أن الأبحاث التي تذكر الاختبار كمرحلة أساسية قليلة نسبيّاً.

في هذا البحث نستعرض آليّة لاستخدام طرائق الأجيل وبنية التطوير المستمر الخاصة بها بهدف بناء خط تطوير مستمر متكامل يتضمّن الاختبار المستمر كمرحلة أساسية فيه، وذلك من أجل اختبار التطبيق على مستويين أساسيين هما الاختبار المؤتمت لواجهة التطبيق البرمجية (API) والاختبار المؤتمت لواجهة المستخدم.

يُظهر هذا البحث أنّ تضمين الاختبار المؤتمت المستمر كمرحلة أساسيّة ضمن خط التكامل/التسليم المستمر يساعد على تقليل الوقت الضائع في الاختبارات اليدوية أو حتى الاختبارات المؤتمتة على بيئات مختلفة. تساعد هذه الخطوة أصحاب المصلحة على الحصول على تغذية راجعة حول التطبيق بمرحلة مبكّرة، وتساعد المطورين على حل

المشاكل التقنية قبل اكتشافها من قبل فريق ضمان الجودة (QA) وبالتالي تفعيل دور الصيانة كمرحلة أساسية من مراحل دورة حياة تطوير المنتجات البرمجيّة.

الكلمات المفتاحية: الاختبار المؤتمت، الاختبار المستمر، أجيل، التّكامل والتّسليم المستمر، اختبار واجهة التطبيق البرمجية، اختبار واجهة المستخدم.

*طالبة دكتوراه في قسم البرمجيات ونظم المعلومات، كلية الهندسة المعلوماتية، جامعة البعث.

** أستاذ في قسم البرمجيات ونظم المعلومات، كلية الهندسة المعلوماتية، جامعة البعث.

*** أستاذة مساعدة في قسم البرمجيات ونظم المعلومات، كلية الهندسة المعلوماتية، جامعة البعث.

Applying Continues Automated Testing in CI/CD Pipeline

Eng. Sally Jarkas*, Dr. Naser Abu Saleh**, Dr. Elida Esber***

Abstract

Due to increased competition and changing markets, companies need flexibility and speed to be aware of the latest updates and take their place in the market. To achieve that, companies rely on new technologies and methods such as Agile and DevOps as a methodology in the company. Although there is a lot of research reviewing the way to achieve agile approaches in applying DevOps ideas, relatively few studies mention testing as a basic stage.

In this research, we proposed a mechanism to use the agile methods and their continuous development structure. The goal is to build an integrated continuous development pipeline that includes continuous testing as a basic stage to test the application at two basic levels, which are automated API testing and automated UI testing.

This research shows how we include continuous automated testing as a key stage within the CI/CD pipeline. The results shows how this mechanism helps reduce time wasted in manual testing or even automated testing on different environments. This step helps stakeholders to get feedback about the application at an early stage, and helps developers to solve technical problems before they are discovered by the Quality

Assurance (QA) team, and as a result, activating the role of maintenance as an essential stage of the software development lifecycle (SDLC).

Keywords: Automated Testing, Continues Testing, Agile, Continues Integration Continues Delivery (CI/CD), API Testing, UI Testing.

* PhD Student, Department of Software Engineering and Information Systems, Faculty of Informatics Engineering, Al-Baath University, Homs, Syria.

** Lecturer, Professor, Department of Software Engineering and Information Systems, Faculty of Informatics Engineering, Al-Baath University, Homs, Syria.

*** Assistant Lecturer, Professor, Department of Software Engineering and Information Systems, Faculty of Informatics Engineering, Al-Baath University, Homs, Syria.

1- مقدمة

لا يمكن لطرق تطوير البرمجيات التقليدية أن تواكب متطلبات العملاء المتسارع**[1]** ، لذلك تم عرض فكرة طريقة تطوير البرمجيات وهي أجيل (Agile) والتي عمدت على إشراك الزبون مباشرة مع فريق التطوير في عملية تطوير التطبيق للحصول على التغذية الراجعة منه بشكل مستمر ومباشر، كما أثبتت الأجيل أنها تمكِّن مطوري البرمجيات من زيادة سرعة العمل، والتكيف بسهولة مع أي تغييرات بكفاءة عالية من خلال إجراء تغييرات وإرسال إصدارات متتابعة للتطبيق. وبناء على المبادئ التي تعتمدها الأجيل ظهرت منهجيات DevOps التي كانت إكمال لمنهجيّات الأجيل بتركيزها على تعزيز العلاقة والتواصل بين فريق التطوير وفريق العمليّات، ولتحقيق هذا التواصل تم اعتماد منهجية التكامل والتسليم المستمر (Continues Integration/Continues Delivery – CI/CD) [2] في العديد من بيئات التطوير البرمجية حيث أن الهدف الأساسي هو

تسهيل التواصل بين الفرق البرمجيّة ضمن الشركة مع زيادة سرعة إيصال المزايا الجديدة للمستخدمين النهائيين وبكفاءة عالية.

تم طرح فكرة التكامل المستمر في عام 2000 [3] وتم تطويرها لاحقاً لتشمل التسليم المستمر أيضاً [4]، وبعد تطبيق هذه الطرائق في التطوير والتسليم تم التوصل إلى أن فائدتها تكمن في تقليل المخاطر الناتجة عن التأخر في التسليم بالإضافة إلى تأمين مؤتمتة مرنة لحل المشاكل التي تظهر في البرمجيات نتيجة التطوير السريع والمتواصل للتطبيقات، وهذا يصب في هدف الوصول لإنتاجية عالية تحقق رضى الزبائن. كل هذه الفوائد حفزت الشركات على الاستثمار وتطبيق التطوير المستمر.

تأتي المشكلة الأساسية أنه وبسبب الإصدارات الكثيرة فإن هناك زيادة في احتمال ظهور الأخطاء، وبالتالي ووفقاً لمبادئ الأجيل فإنه لا بدّ من إعادة التحقق من جودة التطبيق بعد إجراء تعديلات عليه وهذا يستهلك جهداً ومواردا[5] . يأتي الحل في تضمين الاختبارات المؤتمتة المستمرة في عملية توصيل التطبيق للزبائن، حيث يتمّ تطبيق اختبارات سريعة وإعطاء تغذية راجعة مباشرة للزبائن تساعدهم في اتخاذ القرار بإصدار نسخة من التطبيق أم لا.

2- الدراسات السابقة

قامت عدة أبحاث ببناء بيئات عمل لدعم اختبار التكامل المستمر Continues Integration Testing (CIT) حيث قام **[6]** بتوليد حالات الاختبار بشكل مؤتمت جزئياً بالاعتماد على مخطط التتابع كدخل. وقد وضحت نتائج البحث أن بيئة العمل هذه تساعد على عرض الأخطاء حتى قبل ظهورها في بيئة الاختبار.

قام البحث [7] بتطوير منهجية لتقليل الزمن اللازم لتنفيذ الاختبار من خلال إعطاء أولويات لحالات الاختبار التي يتم تنفيذها وبذلك يمكن للمطورين أن يقوموا بالحصول على نتائج الاختبار في مراحل مبكرة من عملية الاختبار. كما تم طرح منهجية اختيار اختبار الانحدار المستمر (Continuous Regression Test Selection –

CRTS) إلى جانب منهجية لتحديد أولوية مجموعات الاختبار المستمرة (Continuous Test Suite Prioritization – CTSP) بهدف تنفيذ اختبارات الانحدار ضمن بيئة التطوير المستمر، حيث يعتمد البحث على استخدام بيانات سجلات تنفيذ مجموعات الاختبار لتحسين زمن تنفيذ الاختبار وتقليل كلفة التنفيذ.

وضح البحث **[8]** أن الشركات تحتاج إلى أن تشرح وتنظم نشاطات وجهود الاختبار قبل الانتقال إلى التطوير المستمر وقاموا بتطوير تقنية التكامل المرئي المستمر (Continues Integration Visualization Technique – CIViT) والتي تهدف إلى عرض نشاطات الاختبار ضمن بيئة التطوير المستمر مما يوفّر الجهود المكررة والمبذولة ويعطي فكرة عن حالة التطبيق بشكل مرئي وفق واصفات اختبار محددة مسبقاً.

يناقش البحث **[9]** موضوع الكم الكبير من البيانات الناتجة عن أدوات التطوير المستمر والتي تشمل الاختبار والتي لا تعطي رؤية واضحة لأصحاب العمل، ولذلك قاموا بإنشاء بيئة عمل ومنصة تدعى SQA–Mashup لدمج وعرض المعلومات الناتجة عن مراحل التطوير المستمر، بالإضافة إلى توفير نموذجين للعرض الأول هو العرض الديناميكي والمناسب للمطورين والمختبرين والآخر هو العرض اللحظي الذي يظهر معلومات عن الأحداث التي تحصل في سلسلة التطوير السريع.

قام البحث **[10]** باستخدام بيئة التطوير المستمر لتطبيق الاختبارات المؤتمتة ضمن بيئة الاختبار للتأكد من صحة سلوك التطبيق البرمجي، حيث تقوم حالات الاختبار بالتحقق من صحة البرمجية بعد كل عملية تعديل على الكود، وعندما يفشل الاختبار فإنه يتوقف إصدار النسخة. أوضح البحث أنه لم يشمل أي اختبار غير وظيفي حيث أنّ المطورين هم المسؤولون عن اختبارات الحمل (Load Testing) غير الوظيفيّة لتحديد ما الذي يجب تطبيقه على كل إصدار.

كان هناك تحدي للاختبار في بيئة التطوير المستمر وهي وجود اختبارات غير موثوقة بما تشمله من عدد كبير للحالات، بالإضافة لتغطية الاختبار المنخفضة، والزمن

الطويل لتنفيذ الاختبار [11] كما ركز البحث على فكرة أن المبرمجين غير قادرين على الحصول على التغذية الراجعة للاختبارات، فعرض فكرة تقسيم خط التكامل المستمر (CI) كحل لهذه المشكلة، ووضح أنه من الصعب التوصل لاختبارات مستقرة وخاصة على مستوى واجهات المستخدم.

هناك العديد من الاقتراحات من أجل تطوير نشاطات الاختبار في خط التطوير المستمر، فقد تم استنتاج أن تطبيق مفهوم التطوير المقاد بالاختبار (Test Driven Development – TDD) والقيام بنشر إصدارات يومية من التطبيق هي ممارسات أساسية في بيئة التطوير المستمر، كما تم طرح فكرة أن واحد من الأمور التي تقلل الاختبار اليدوي في بيئة التطوير المستمر هو القيام بالتخطيط للاختبار وهذا يتطلب تعاوناً بين فريق التطوير وفريق الاختبار لتحديد وتوثيق لائحة بحالات الاختبار المهمة التي يجب تحويلها كمؤتمتة**[**12**]** .

قام البحث **[13]** بتطوير بيئة اختبار مستمر لواجهة التطبيق ضمن خط التطوير المستمر وأوضح البحث أهمية والفوائد الكثيرة لتطبيق الاختبارات المستمرة. لم تكتفي الأبحاث بتطبيق الاختبار الوظيفي فقد قام **[**14**]** ببناء بيئة اختبار تقوم على تطبيق اختبار الحمل ضمن مراحل التطوير المستمر واستخدم عدد من الأدوات كما قام بعرض نتائج تعطي تحليل سلوك التطبيق في كل المراحل مع إمكانية القيام بتوسيع اختبار الحمل ليشمل أعداد مختلفة من المستخدمين.

3- أهداف البحث

انطلاقاً من التوصيات ونتائج الدراسات السابقة نلخص أهداف البحث بالنقاط الآتية:

- دراسة بيئة التطوير البرمجية أجيل وتوضيح مفهوم DevOps بشكل سريع
- تطبيق الاختبار المستمر كمرحلة من مراحل التطوير المستمر من خلال

- إعداد بيئة مناسبة لتطبيق الاختبار المؤتمت لواجهة التطبيق البرمجيّة ضمن خط التطوير المستمر.
- إعداد بيئة مناسبة لتطبيق الاختبار المؤتمت لواجهة المستخدم ضمن خط التطوير المستمر.
- إعداد آلية لتطوير تقارير بنتائج الاختبار وتوصيلها إلى أصحاب العمل والمطورين.

4- موارد وطرق البحث

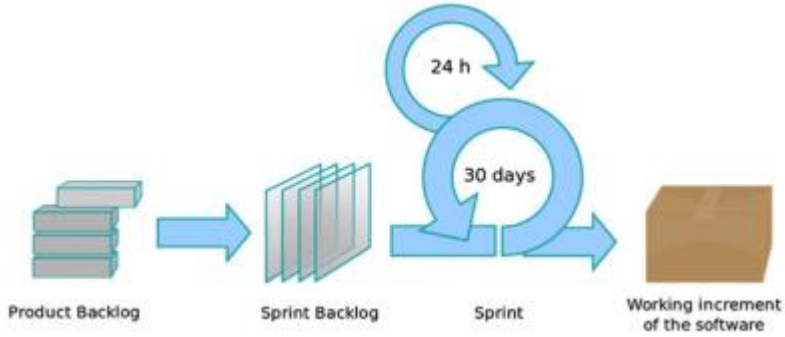
يبدأ البحث بتعريف منهجية تطوير الأجيل كمنهجية حديثة من مناهج تطوير البرمجيات، ثم يأتي البحث على عرض مفصّل لأساليب ال DevOps والأساليب المُتّبعة فيها للتطوير المستمر والتي تشمل التكامل، التسليم والنشر المستمر. وبما أن هدف البحث الأساسي هو عرض الاختبار المستمر فقد قام البحث بتقديم هذا المفهوم وتوضيحه بعد عرض سريع لمراحل الاختبار. كما تم بعدها عرض بعض من أدوات التطوير المستمر، لننتقل بعدها إلى الدراسة التجريبيّة التي قمنا فيها بإعداد وبناء بيئة اختبار وتضمنيها ضمن خط تطوير التطبيق البرمجي وتضمن مراحل عمل البيئة البرمجية التالي:

1. تجهيز سيناريوهات اختبار واجهة التطبيق البرمجيّة مع كتابة سكريبت الاختبار الخاص بها وتصديرها.
2. إعداد بيئة الاختبار المناسبة ضمن خط التطوير لتضمين وتنفيذ اختبارات واجهات التطبيق وإصدار تقارير بالنتائج.
3. ربط اختبارات واجهات المستخدم المؤتمتة مع مستودع الكود البرمجي
4. إعداد بيئة مناسبة للاختبارات المؤتمتة لواجهات المستخدم تتضمن تأمين واجهة مناسبة لأصحاب العمل حيث يسهل عليهم تنفيذ الاختبار والحصول على تقارير مفصلة عن الأخطاء البرمجية.

5- منهجية تطوير الأجيل (Agile Software Development)

هي طريقة من طرق تطوير البرمجيات هدفها الأساسي هو تسليم منتجات برمجية بسرعات عالية مهما كان التّغيير في هذه المنتجات باعتمادها على التنسيق العالي بين أفراد الفريق، تركز منهجية الأجيل على مجموعة مبادئ أساسية منها السرعة، التخطيط والعمل المستمر، إشراك الزبون كعضو أساسي في فريق التطوير، والتكيف بكفاءة مع متطلبات السوق المتغيرة. **[4]**

الأجيل هي منصة لتطوير تطبيقات بجودة عالية من خلال تجميع وتحقيق متطلبات الزبون بأسلوب مرن. فكما يشير الشكل (1) يتم تجميع المزايا (Features) التي سيتم تطويرها ضمن مستودع تراكم المنتج (Product Backlog) من خلال فريق منظم ذاتيّاً وملتزم بمبادئ الأجيل. في المرحلة التالية يتم البدء بالتطوير البرمجي لمجموعة مزايا مُختارة من مستودع تراكم السبرنت (Sprint Backlog) حيث يوافق الفريق في بداية كل سبرنت على المزايا التي يختارها فريق التحليل ليتم تطويرها وتسليمها في الموعد المحدد بعد أن يتم اختباره. يتم العمل على بناء أي ميّزة جديدة بشكل متزايد وبأطوار تكرارية تدعى سبرنت (sprint)، حيث يتم تحديد مدة كل سبرنت بين اثنان إلى 4 أسابيع. بعد اختبار الميّزة الجديدة والموافقة عليها يتم إصدارها للزبائن. يتطلب بناء منتج برمجي متكامل عدداً لا بأس به من السبرنت، لذلك ينصح بأن يكون الإصدار صغير قدر الإمكان ويحوي المتطلبات ذات القيمة الأعلى حالياً بالنسبة للزبون حتى يتم استكمالها بشكل متزايد.



الشكل 1- مراحل تطوير منهجية الأجيل

تعطي التغذية الراجعة التي يحصل عليها الفريق عند نهاية كل سبرنت الثقة بتوصيل التطبيق للزبائن، وبهذا يكون الاختبار المطبق في بيئة التطوير أجيل هو الحل الأفضل للتطبيقات المنتشرة حالياً والتي تعتمد على المتطلبات المتغيرة وعوامل البيئة المتغيرة.

إنّ واحد من أهم مبادئ الأجيل هو "إيصال نسخة من التطبيق للزبون بشكل مستمر وجودة عالية" وهنا ظهرت الحاجة إلى التركيز على دمج العمليّات وتنظيم أدوار فريق التطوير لتحقيق أهداف المشروع وتمّ ذلك بتطبيق ثقافة ال **DevOps** التي تعمل على تقديم مبادئ لإنتاج وتسليم المنتجات بشكل مرن وسريع. **[15]**

6- التكامل والتسليم المستمر (Continues Integration/Continues Delivery-CI/CD)

لكي يتم تطبيق مفاهيم الأجيل بسهولة ومعرفة المهام في بيئة التطوير هذه، تمّ اقتراح تعريف ال **DevOps** الذي يتم فيه التركيز على تعزيز العلاقة بين الفرق البرمجيّة وأساساً فريقي التطوير وفريق العمليّات، واعتبار ال **DevOps** بأنه ممارسة من ممارسات تطوير الأجيل تستخدم أفكار وعمليات وتقنيات وأدوات لتبسيط واختصار زمن دورة حياة البرمجيات بهدف تسريع عملية تسليم المنتج بسرعة وكفاءة عالية، والتكيف

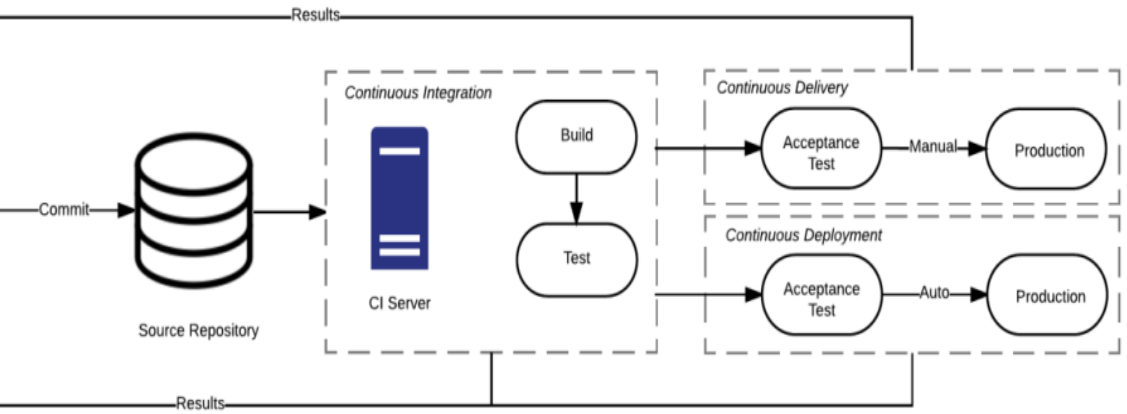
بسهولة مع إصلاح الأخطاء وإضافة مزايا جديدة للتطبيق بشكل دوري ومستمر، مما يسهل الانتقال من العمل اليدوي المكلف إلى العمل المؤتمت.

لكي يقوم فريق ال DevOps بتحقيق أهداف ومبادئ منهجية تطوير الأجيل عمليّاً فإنه يستخدم مجموعة من تسلسل أعمال(**workflow**)، والممارسات ومبادئ التشغيل التي يتمّ اتّباعها وتُعرف بالتكامل المستمر/التسليم المستمر حيث يتمّ استخدام الأدوات المؤتمتة وأدوات الاختبار الصحيحة ضمن خط التكامل والتسليم المستمر (CI/CD Pipeline)، والذي سنسميه اختصاراً في البحث بخط التطوير، لتسليم المنتج النهائي بشكل متزايد مما يساعد الفريق على اكتشاف الأخطاء بمراحل مبكّرة ممّا يُخفِّض الجهود والتكاليف، وذلك بما للوصول إلى هدف تحقيق مبادئ منهجية التطوير أجيل، وفيما يلي نستعرض بوضوح العمليات في خط التكامل والتسليم المستمر.

1- **التكامل المستمر** (Continues Integration- CI): هو ممارسة من ممارسات تطوير البرمجيات يقوم فيها المطورون بدمج الكود الرئيسي الخاص باستمرار عدة مرات في اليوم في مستودع مشترك وفقاً لسياسة معينة. يتيح التكامل المستمر للمطورين أن يقوموا بتطوير أفرع مختلفة من الكود البرمجي والتي يمكن دمجها لاحقاً بشكل أتوماتيكي عند التأكيد على أخذ هذه التغييرات البرمجية بعين الاعتبار. يعتبر استخدام نظام تحكم بالإصدار (Version Control System – VCS) هو أحد مكونات تحقيق إجرائية التكامل المستمر لأهميته في إنشاء إصدارات التطبيق دون الحاجة لوجود مكتبات أو اعتمادات خارجية. يتمّ استخدام بنية تلقائية للتحقق من جودة كل تحديث يتم دفعه إلى المستودع، مما يتيح للمطورين تحديد المشكلات وحل الأخطاء بسرعة.

2- **التسليم المستمر** (Continues Delivery- CD): هي العملية التي يتم فيها التأكد من أن الكود البرمجي (بما فيه من ميّزات جديدة، إصلاح أخطاء) قابل للنشر والتسليم في أي وقت تحت طلب الزّبون وبشكل روتيني بسيط. يتمّ ذلك عن طريق تغليف ونشر ما تقوم به مرحلة التكامل المستمر باستخدام أدوات معينة لبناء واختبار ونشر التطبيقات بمرحلة واحدة لتقديم متطلبات الزبائن كتطبيقات بجودة عالية بشكل آمن وسريع. ويمكن أن يُشار إلى هذه المرحلة باسم **النشر المستمر** (Continues Deployment) حيث أنها العملية التي تتم فيها إدارة الكود بدءاً من المرحلة التي يقوم فيها المطور بإرسال تغييرات الكود إلى نظام تحكم الإصدار ومن ثم إجراء عمليات التكامل، وصولاً لبناء التطبيق على بيئة عمل المستخدمين بدون الحاجة إلى وجود جهد يدوي حيث أن كامل العمليات تكون مؤتمتة بدءا من التطوير حتى الإصدار. إن عملية التسليم النهائي في مرحلة التسليم المستمر هي عملية يدويّة يقرّر فيها الزبون ما الذي سيتم نشره ومتى، على عكس النشر المستمر التي تكون عملية مؤتمتة وبالتالي فهي مرحلة مكمّلة للتسليم المستمر.

يوضح الشكل (2) العلاقة بين عمليات التطوير المستمر المذكورة سابقاً، حيث أنه وبعد الحصول على الكود البرمجي المطلوب من مختلف المطورين يتم إجراء عملية تكامل مستمرة على سيرفر محدد تتضمن بناء واختبار الإصدار وبعد نجاح هذه الخطوة فإنه يتم على التوازي تحقيق التكامل والتسليم المستمر لإيصال التطبيق إلى المستخدمين.



الشكل 2 – العلاقة بين طرق التطوير المستمر

عملية النشر المستمر تتضمن عملية التسليم المستمر ولكن العكس غير صحيح، ويجب على النشر المستمر أن يكون مؤتمت بشكل كامل على خلاف التسليم المستمر الذي من الممكن أن يكون يدوي ويكون العمل اليدوي هو في مرحلة اتخاذ قرار النشر من قبل محللي النظام.

2-6 أدوات التكامل/التسليم المستمر

هناك عدد من الأدوات التي يمكن استخدامها لتطبيق التكامل المستمر/التطوير المستمر CI/CD نذكر منها

1- CircleCI
2- TeamCity
3- Bamboo
4- GitLab
5- Buddy
6- Travis CI
7- Bitbucket pipelines
8- Jenkins

سنأتي على شرح آخر أداتين بالتفصيل لأنهما الأداتان المستخدمتان في هذا البحث. حيث أنّ ما يهمنا هو إعداد كل أداة من الأداتين للاستفادة من خدماتها وتطبيقها على الاختبارات المؤتمتة.

1- **Bitbucket pipelines**: وهي منصة إدارة للكود تتضمن خط التطوير (Pipeline) بالإضافة لأنها خدمة CI/CD متكاملة مدمجة تسمح بعملية بناء واختبار ونشر الكود البرمجي بالاعتماد على ملف إعداد YAML داخل مستودع الكود. حيث يتضمن هذا الملف التعليمات التي تقوم بكلّ مما يلي:

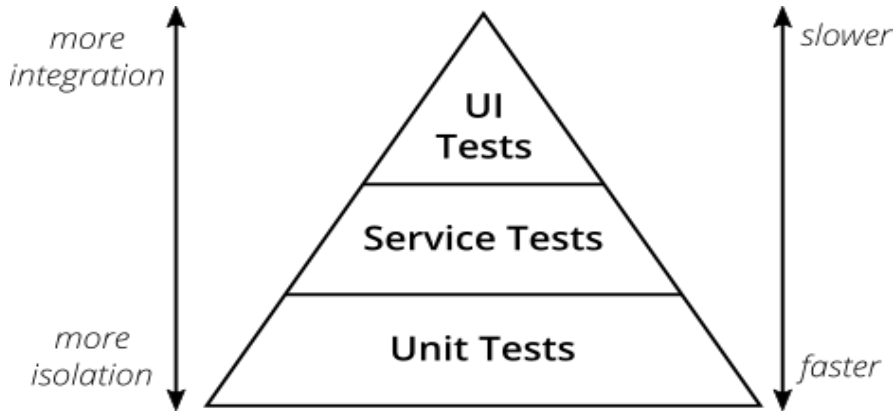
- فحص أكواد الاختبار فيما إذا كانت تحوي على تغييرات.
- ترجمة الكود إلى اللغة المصدرية.
- تنفيذ اختبارات الوحدة.
- إنشاء مكتبات خاصة بالنشر.

2- **Jenkins**: وهي واحدة من أهم الأدوات الموجودة في السوق مفتوحة المصدر، ما يميزها أنها تتمتع بواجهة استخدام مناسبة للمبتدئين وسهلة الإعداد لتناسب أي بيئة عمل. تدعم هذه الأداة استخدام إضافات (Plugins) مختلفة عند الحاجة، كما تؤمن سرية عالية لخط التطوير(Pipeline)، بالإضافة إلى الإمكانية التي تعطيها هذه الأداة لإدارة وعرض التقارير وجدولة أوامر التنفيذ. تتمحور آلية عمل ال Jenkins بشكل عام بأنه يتم استدعاء رابط webhook من مستودع الكود (git Repository) عند حصول أي تغيير في هذا المستودع، وبالتالي تقوم الأداة بالحصول على هذه التغييرات وتنفيذ مجموعة من الإعدادات التي تم إدخالها عند تجهيز إعدادات البيئة. وهنا تختلف النتائج حسب الإعدادات.

7- الاختبار

هناك العديد من مراحل الاختبار وكل مرحلة تتطلب طرائق وأدوات مختلفة للتنفيذ اعتماداً على المجال الذي يتم اختباره، ينقسم هرم الاختبار الموضح في الشكل (3) إلى 3 طبقات وهي اختبار الوحدة، اختبار الخدمة واختبار الواجهات. **[7]**

يُعبّر اختبار الوحدة عن تطبيق الاختبار على وحدات أو أجزاء صغيرة من النظام بحيث يتم ضمان أن هذا الجزء يعمل بالشكل الصحيح ويتم تحقيق هذا الاختبار من قبل المبرمجين وعلى مستوى الكود (ليس ضمن مجال البحث). يتم التحقق في اختبار الخدمة من خدمات التطبيق بأنها تؤدي الغرض المطلوب من حيث الوظيفة، الموثوقية، والأداء، في حين يقوم اختبار واجهات المستخدم بالتحقق من أن التطبيق يطابق متطلبات الزبون الوظيفيّة.



الشكل 3 – هرم الاختبار

كما هو واضح من الشكل (3) فإنّ هذه الهرمية تقترح بأن يتم إنشاء عدد كبير من حالات الاختبار في قاعدة الهرم (اختبارات الوحدة) لما تتمتّع به من سرعة في التنفيذ باعتبار بأنه يتم تنفيذه على مستوى الكود وتقليل عدد حالات الاختبار كلما تم الانتقال إلى مستوى أعلى في الهرمية.

7-1 الاختبار المستمر

لم يكن هناك تعريف واضح للاختبار المستمر وإنما كان يعبر فقط عن تنفيذ اختبار الوحدة (Unit Test) على بيئة عمل المطورين، لكن مع ازدياد أداوت الاختبار المؤتمت فقد توسع تعريف أنواع الاختبار وتوسع تعريف الاختبار المستمر بناء عليها وأصبح بأنه عملية تنفيذ حالات الاختبار المؤتمت بسرعة عالية لتزويد المهتمين سواء كانوا مبرمجين أو زبائن بالتغذية الراجعة عن المخاطر المحتملة في التطبيق في مراحل مبكرة قدر الإمكان وقبل نشر التطبيق للمستخدمين.**[7]** يتم تحقيق الاختبار المستمر من خلال جعل الاختبار المؤتمت كمرحلة ضمنية في خط التطوير.

تتضمن عملية الاختبار المستمر في بيئة التطوير أجيل القيام بتنفيذ الاختبارات بشكل مبكر ومتكرر على كل أو بعض أجزاء التطبيق وفي أوقات مختلفة. يمكن أن يشمل الاختبار المستمر كافة مراحل الاختبار المذكورة في هرم الاختبار **[8]**، كما يمكن تطبيق اختبارات الانحدار للتأكد من أن التغييرات المستمرة التي تحصل في بيئة التطوير أجيل لا تؤثر على التطبيق بشكل كبير. يمكن لكافة أنواع الاختبارات هذه أن يتم تنفيذها بشكل مؤتمت من خلال كتابة أكواد الاختبار التي توفر الوقت اللازم للقيام بالاختبار يدويّاً وتؤمن كفاءة ودقة عالية لنتائج الاختبار المتكرر في بيئة تطوير الأجيل.

8- الدّراسة التجريبية

تكمن المشكلة الأساسية في قلّة الأبحاث والدّراسات الموجودة في مجال تطوير الاختبار المستمر، حيث لا تذكر الأبحاث تطبيق بيئة متكاملة تراعي الزيادة الكبيرة في الأخطاء الناتجة عن تطبيق منهجية الأجيل ضمن خط التطوير المستمر، بالإضافة لوجود كم كبير من بيانات الاختبار التي لا تعطي صورة مجردة لأصحاب العمل عن نتائج الاختبار النهائية. لذلك توجه البحث إلى القيام بتطبيق الاختبار المستمر الذي يشمل مستويين أساسيين من مستويات الاختبار وهما اختبار واجهات التطبيق البرمجيّة (API) واختبار واجهات المستخدم (UI) وفق المراحل التالية:

1. القيام بإنشاء اختبارات واجهة التطبيق وتضمينها ضمن خط التطوير.
2. القيام بتضمين اختبار واجهات المستخدم في بيئة تطوير مستمر.
3. إصدار تقرير بالنتائج في كلا المستويين.

8-1 البيئة المدروسة على الحالة الدراسية (Case Study)

سنقوم بإجراء الاختبارات على تطبيق برمجي يمثّل نظام إدارة علاقات العملاء والمذكور في كل من البحثين [19] [20]. يشمل الاختبار مستويين أساسيين، يُمثِّل المستوى الأول الاختبار الوظيفي المؤتمت لواجهات المستخدم والذي تمّ تطبيقه في البحث [19] ويُمثِّل المستوى الثاني الاختبار المؤتمت لواجهة التطبيق البرمجيّة المذكورة في البحث [20].

8-2 إعداد الاختبار المؤتمت لواجهات التطبيق البرمجيّة ضمن خط التكامل /التسليم المستمر

الهدف الأساسي من هذه الخطوة هو تحقيق مستوى من مستويات الاختبار وهو اختبار الخدمات وتحديداً اختبار واجهات التطبيق البرمجيّة، لذلك قمنا بالاعتماد على نتائج البحث [20] الذي ركّز بشكل أساسي على إنشاء حالات الاختبار وتقليلها قدر الإمكان. قمنا باستخدام مجموعات الاختبار المذكورة في البحث وإنشاء سكريبتات وسيناريوهات اختبار واجهة التطبيق البرمجيّة عن طريق كتابة مقاطع برمجية يتم فيها التأكد من أن هناك استجابة من السيرفر وأنه لا توجد أخطاء وأن الاستجابة التي تم إعادتها هي من نوع صيغة json. وهذا موضح في المقطع البرمجي الموضح في الشكل (4).

```
pm.test("test user loggin in", function() {
    var data = pm.response.json()['data'];
    pm.environment.set("admin_token", data.user.token);
});

 pm.test("response should be okay to process", function () {
    pm.response.to.not.be.error;
});

pm.test("response must be valid and have a body", function () {
     pm.response.to.be.withBody;
     pm.response.to.be.json;
});


pm.test("test admin login", function() {
   pm.response.to.have.status(200);
   var data = pm.response.json()['data'];
});
```

الشكل 4 – مقطع برمجي لسكريبت اختبار جزء من واجهة التطبيق البرمجيّة

قمنا في المرحلة التالية بكتابة الشكل العام (schema) والتي يجب أن تكون مطابقة لاستجابة الطلبات (Responses)، حيث تحوي ال (schema) على تعريف بأنماط الحقول والأنماط الجزئية، والتحقق فيما إذا كانت هذه الحقول مطلوبة في جميع الاستجابات أم أنها اختيارية كما يتم التحقق من مطابقة حقول استجابة الطلب العائدة مع الشكل العام لهذا الطلب، يوضح الشكل (5) أحد ال schema الخاصة بتسجيل دخول المدير إلى التطبيق.

```
var Ajv = require('ajv'),
    ajv = new Ajv({logger: console})
    //schema=JSON.parse(pm.environment.get('Customerschema'));

    AsminLoginschema = {
      "properties": {

        "data": {
            "type": "object",
            "properties": {
                "user":{
                    "properties":{
                        "id": {"type": "number"},
                    "first_name": {"type": ['null', 'string']},
                    "last_name": {"type": ['null', 'string']},
                    "email": {"type": ['null', 'string']},
                    "phone_number": {"type": ['null', 'string']},
                    "type": {"type": ['null', 'string']},
```

الشكل 5 – يوضح ال schema الخاصة بتسجيل دخول للمدير إلى التطبيق

قمنا في الخطوة التالية بإنشاء سيناريوهات اختبار وتنفيذ الاختبار المؤتمت لواجهات التطبيق البرمجيّة ويشمل التحقق من مطابقة الخرج الناتج للخرج الفعلي وهذه

الخطوة هي التي تحدد نجاح الاختبار أو فشله وهذا موضح في الشكل (6) وهكذا نكون قد انتهينا من إعداد الاختبارات الخاصة بواجهة التطبيق البرمجيّة.

```
pm.test('No errors are returned', function() {
    var error = pm.response.json()['errors'];
    var validate = ajv.compile(AsminLoginschema);
    pm.expect(error).to.be.false;
```

الشكل 6 – يوضح سكريبت الاختبار الذي يقارن الخرج الناتج مع الخرج الفعلي

في المرحلة التالية قمنا بتصدير سكريبتات وشروط الاختبار كمجموعة (Collection) وإرفاقها في المشروع البرمجي. وفيما يلي سنقوم بتوضيح العملية التي تم فيها إعداد بيئة التكامل/التسليم المستمر ليتم تنفيذ اختبارات واجهة التطبيق البرمجيّة ضمنها.

قمنا بتجهيز إعدادات تهيئة بيئة اختبار الخدمة ضمن ملف pipelines.yml وإنشاء خطوة (step) في ملف الإعداد المستخدم لتجهيز CI/CD pipeline، تتضمن عملية تجهيز خط التكامل/التسليم المستمر القيام بتحديد التعليمات الخاصة لتنفيذ الاختبار ضمنه.

تتضمن التعليمات الخاصة بإرفاق الاختبارات داخل خط التطوير أن يقوم المختبر بتحديد مسار مجموعة الاختبار التي قمنا بتصديرها سابقاً، بالإضافة إلى تحديد مسار بيئة الاختبار الذي تم تصديره مسبقاً بنفس الآلية التي قمنا بها عند تحديد مجموعة الاختبار. في المرحلة التالية نقوم صراحةً بذكر أسماء مجموعات اختبار الخدمة التي نرغب في تنفيذها ضمن خط التطوير، مع العلم أن هناك إمكانية لتنفيذ كافة مجموعات الاختبار من خلال استخدام رمز النجمة *. بهذه الخطوات قمنا بتضمين الاختبارات ضمن خط التكامل/التسليم المستمر والشكل (7) يوضح سكريبت تنفيذ اختبار واجهات التطبيق ضمن خط التكامل/التسليم المستمر.

```
step:
    name: newman runner
    image: postman/newman
    script:
        - npm --version
        - newman --version
        - >
          newman run ./newman/PostmanIntegrationTestJourneys.json
          -e ./newman/stagingEnvironment.json
          -g ./newman/TotersTeamWorkspace_globals.json
          -r cli -k --insecure --delay-request "1500"
          --folder "Terms and Conditions"
          --folder "OnLoad"
          --folder "Setup"
          --folder "home page "
          --folder "Stores"
          --folder "Groceries "
          --folder "My Cart"
          --folder "Active Orders"
```

الشكل 7 – خطوات تضمين اختبار واجهة التطبيق البرمجيّة ضمن خط التطوير المستمر

عندما يقوم المبرمجون بإضافة أي كود برمجي يتم تنفيذ الخطوات الموجودة في خط التطوير ومن ضمنها خطوة الاختبار، فعند الوصول لمرحلة الاختبار لدينا حالتان الأولى تتمثل بنجاح الاختبار وبناء على هذه النتيجة ينتقل خط التطوير لتنفيذ الخطوة/ات التالية وصولاً إلى مرحلة نشر التطبيق ضمن بيئة عمل المستخدم، بينما تتمثل الحالة الثانية بفشل الاختبار وهنا سيظهر الخطأ وتتوقف خطوات التنفيذ عند هذه المرحلة.

في المرحلة اللاحقة استكملنا العمل وذلك بإضافة خطوة لتوليد التقارير لتوفر لمسؤولي التطبيق السهولة في الاطلاع على الأخطاء الظاهرة، قمنا بتحقيق هذه الخطوة من خلال التعليمة التالية التي تتضمن توليد تقرير ويب من نوع HTML ويوجد خيارات لتوليد تقارير من أنواع مختلفة تتضمن إنشاء تقارير من نوع xml أو junit.

-r cli,htmlextra --reporter-cli-no-console --reporter-htmlextra

وهكذا نكون قد حققنا أول خطوة من خطوات إنشاء خط تطوير مستمر قابل للتطبيق. كما يجب أن نذكر أن هذه الإعدادات قابلة للتطبيق على أي مجموعة اختبار واجهة تطبيق برمجيّة ويمكن تعميمها، حيث أنها لا تتعلّق بنوع التّطبيق ويكفي استبدال أسماء مجموعات الاختبار. حيث يوضّح الشكل (8) عمليّة تضمين خطوات اختبار

واجهة التطبيق البرمجيّة ضمن خط التطوير المستمر لأحد تطبيقات إدارة علاقة العملاء حيث تمّ الاكتفاء بذكر أسماء مجموعات الاختبار دون أي تغيير في بيئة العمل أو إعادة العمل.

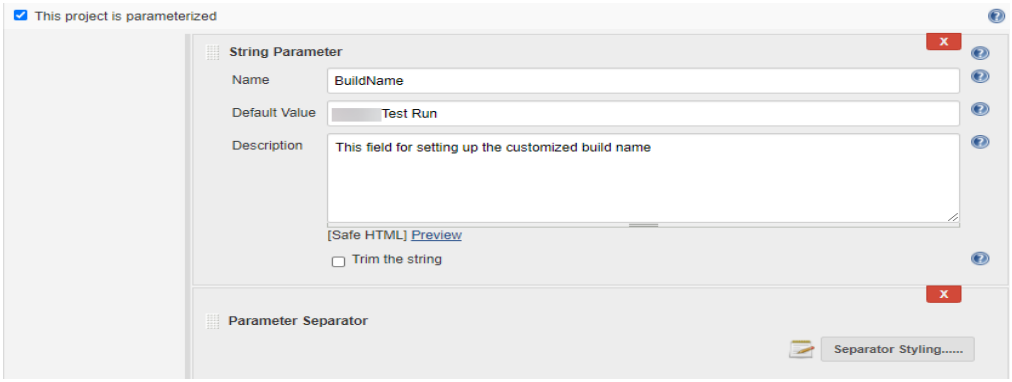
```
- step:
        name: newman runner
        image: postman/newman
        script:
                - npm --version
                - newman --version
                - >
                  newman run ./newman/ERPTestJourneys.json
                  -e ./newman/stagingEnvironment.json
                  -g ./newman/TotersTeamWorkspace_globals.json
                  -r cli -k --insecure --delay-request "1500"
                  --folder "Login"
                  --folder "Adminstration"
                  --folder "Sales"
                  --folder "StaffMgt"
                  --folder "ClientMgt"
                  --folder "Operators "
                  --folder "Accounting"
```

الشكل 8 – خطوات تضمين اختبار واجهة التطبيق البرمجيّة ضمن خط التطوير المستمر لتطبيق برمجي مختلف

8-3 إعداد الاختبار المؤتمت لواجهات المستخدم ضمن خط التكامل /التسليم المستمر

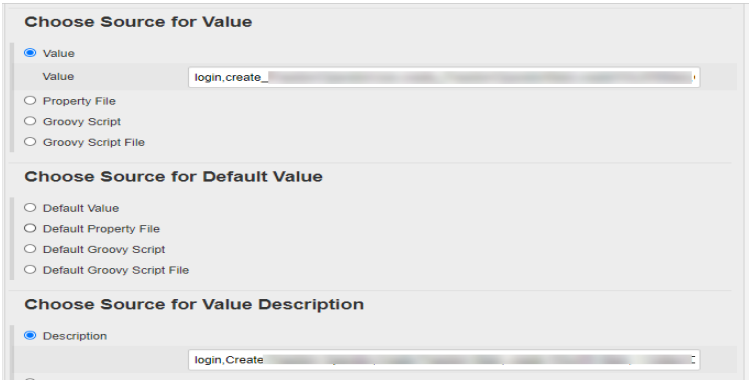
في هذه الخطوة سنقوم بإعداد بيئة اختبار مؤتمت مستمر تستهدف اختبار واجهات المستخدم على نفس التطبيق البرمجي المستخدم في الفقرة 8-2، وسنستفيد في هذه المرحلة من الاختبارات المؤتمتة المذكورة في البحث [19] لتنفيذها ضمن خط التكامل/التسليم المستمر.

بداية يتم التأكد من أن كود الاختبار موجود في مستودع الأكواد، ثم نقوم بربط هذا المستودع مع بيئة خط التكامل/التسليم المستمر. من المهم التأكد من تفعيل خيار تنفيذ الاختبار عند أي عملية تغيير في الكود البرمجي في المستودع. الهدف من الخطوات القادمة هو القيام بإضافة إعدادات تسمح لمستخدمي بيئة الاختبار من تحديد حالات الاختبار المُراد تنفيذها من واجهة الأداة. ولتحقيق ذلك قمنا أولاً بإضافة خيار بأن يكون المشروع قابل لقراءة أي مدخلات مع إضافة خيار للمستخدم بأن يقوم بكتابة اسم الاختبار، كما يوضح الشكل (9).



الشكل 9 – إعدادات إضافة اسم ووصف اختياري لمرحلة تنفيذ الاختبار

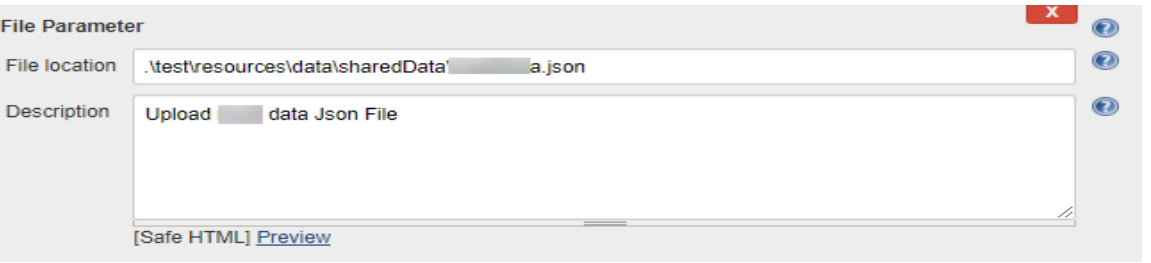
بعدها قمنا بتجهيز الإعدادات اللازمة للمستخدم ليتم عرض قائمة منسدلة له تمكّنه من اختيار حالات الاختبار التي يرغب بتنفيذها من خلال ربط الاسم البرمجي (الذي يظهر في القسم الأول Source Of Value) لسيناريوات الاختبار مع الاسم الذي يظهر للمستخدم (Source Of Value Description)، وهذه الإعدادات موضحة في الشكل (10).



الشكل 10 – إعدادات إنشاء قائمة بحالات الاختبار المتوفرة

بناء على ما ذُكر في [19] فإنّ حالات الاختبار المؤتمتة تعتمد على مفهوم القيادة بالبيانات (Data Driven)، لذلك كان لا بد أن نتيح الإمكانية للمستخدمين أن يقوموا بإرفاق ملفات البيانات المرغوب بها والتي ممكن أن تكون بصيغ مختلفة json, xsls. اعتمد البحث على استيراد الملفات من نوع json كمصادر لبيانات الاختبار،

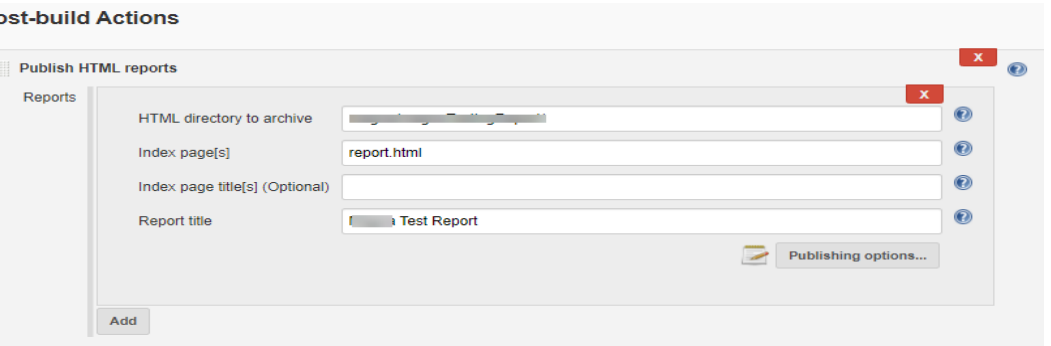
ودعمنا ذلك في بيئة الاختبار من خلال إضافة الإعدادات اللازمة لذلك. يبيّن الشكل (11) الإعدادات الخاصة بإعداد الواجهة الخاصة بإرفاق ملفات بيانات الاختبار.



الشكل 11 – إعدادات إرفاق ملفات البيانات للاختبار خارجياً

هكذا نكون قد انتهينا من تهيئة الإعدادات السابقة لعملية الاختبار، في الخطوات التالية سنقوم بعرض الإعدادات التي قمنا باستخدامها كمرحلة لاحقة للاختبار، التي تتمثل في تصدير تقارير الاختبار وإرسال إيميل للمعنيين بالتقرير المطلوب.

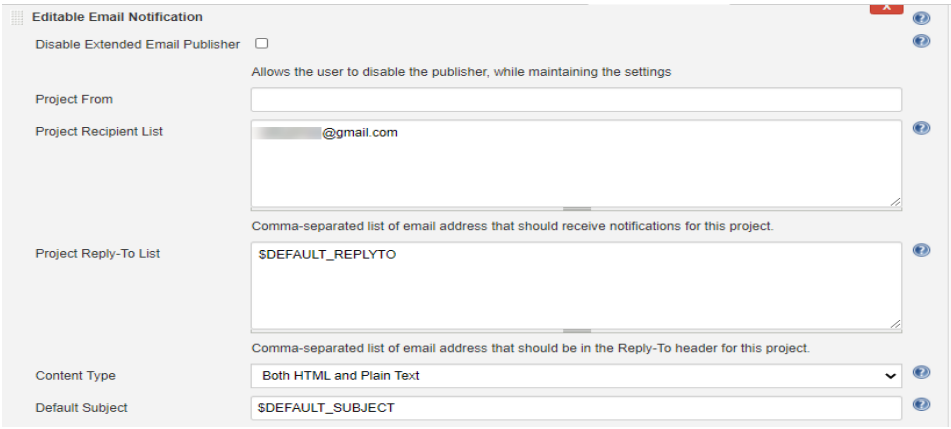
تأتي أهمية التقارير من أهمية الاختبار لما تعطيه من تفاصيل دقيقة عن نتائج الاختبار دون الخوض في تفاصيل الكود البرمجي والعمليات البرمجية وخاصة لغير المهتمين، حيث نضيف إلى الإعدادات مسار الملف المصدر لنتائج الاختبار بالإضافة إلى اسم التقرير، وأيضاً المسار الذي سيتم تصدير ملف تقرير الاختبار إليه. يوضح الشكل (12) الإعدادات التي قمنا بها لإضافة التقارير كخرج لعملية الاختبار.



الشكل 12– إعدادات عملية توليد تقارير الاختبار المؤتمت

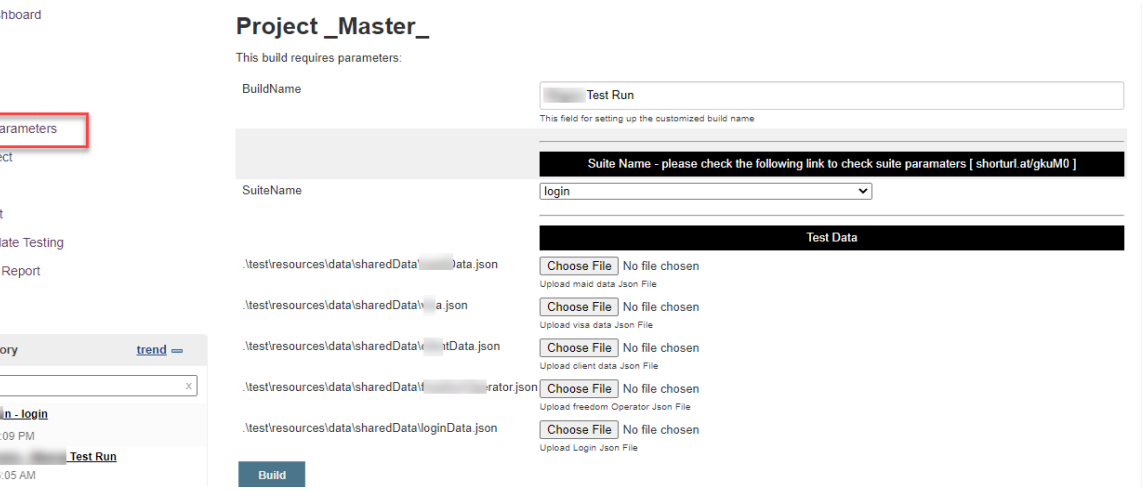
تُظهر الخطوة الموضحة في الشكل (13) إعدادات إرسال بريد الالكتروني إلى الأشخاص المهتمين بنتائج الاختبار المؤتمت، يحث يتم إضافة قائمة الأشخاص التي

ستستقبل البريد الالكتروني الذي يحوي معلومات الاختبار، كما يتم إضافة عنوان ونوع محتوى هذا البريد.



الشكل 13 – إعدادات إرسال بريد الكتروني كمرحلة لاحقة للاختبار

بالنتيجة وبعد تهيئة الإعدادات فإن نتيجة عملنا هو إبعاد المسؤولين عن التطبيق البرمجي عن تفاصيل الكود البرمجي وتأمين واجهة سهل أن يتعامل المستخدمون معها، كما هو موضح في الشكل (14)



الشكل 14 – واجهة المستخدم الأساسية بعد الانتهاء من تهيئة الإعدادات

كما نلاحظ في الشكل (14) فإن هذه الواجهة سهلة الاستخدام لأي مستخدم مختص أو غير مختص لما توفره للمستخدم من سهولة بكتابة اسم الاختبار مع وجود قيم

افتراضية، بالإضافة إلى وجود قائمة منسدلة تمكّن المستخدم من اختيار حالة الاختبار التي يريد أن يقوم بتنفيذها بالإضافة إلى وجود الإمكانية للمستخدم أن يقوم بإضافة ملف الاختبار الخاص فيه.

كما نلاحظ فإننا بإعدادنا لبيئة الاختبار هذه فإن هذه الإعدادات عامّة ومناسبة لأي اختبار واجهة خاص بأي تطبيق برمجي، حيث أن الاختلاف فقط يكون في أسماء مجموعات الاختبار كما ذكرنا في اختبار واجهة التطبيق البرمجيّة. حيث أنّ هذه البيئة مستقلة عن التطبيق.

9- النتائج والمناقشة

قمنا في هذا البحث بعملية تضمين الاختبارات المؤتمتة في خط التكامل /التسليم المستمر (CI/CD) سواء كانت الاختبارات المؤتمتة لواجهة التطبيق البرمجيّة أو لواجهات المستخدم. تميّز عملنا في البيئة المحددة عن الدراسات السابقة:

1- إنّ الأبحاث **[14] [13] [12] [11] [10] [7]** ركّزت على تطبيق اختبار واجهة المستخدم واختبار الحمل والتطبيق المُقاد بالاختبار كلّ منها بشكل منفصل. بينما قمنا في البحث بإضافة اختبار واجهة التطبيق البرمجيّة كمرحلة أساسية من مراحل التطوير المستمر والتي تشمل أيضاً اختبار واجهة المستخدم، وهذا ما أضاف على الأبحاث السابقة مستوى آخر من مستويات الاختبار.

2- تمّ إضافة إعدادات لإيقاف تنفيذ خط التطوير المستمر في حال وجود خطأ اختبار في أي مستوى من المستويات، في حين يتم متابعة المرحلة اللاحقة في حال نجاح تنفيذ المرحلة الحاليّة.

3- لقد قمنا بتطوير لما قدّمته الأبحاث**[9] [8]** في عرض نتائج الاختبار، فقد قمنا بإضافة عملية إرسال التقارير عبر البريد الالكتروني لكافة المستخدمين المسجلين عبر خط التطوير المستمر كالمطوّرين، يتمّ إرسال التقرير في حال نجاح/فشل الاختبار ويحوي معلومات تفصيليّة كما هو موضح سابقاً.

4- قام البحث [7] بتقليل زمن تنفيذ الاختبار من خلال إعطاء أولويّة للاختبارات، بينما قمنا في بحثنا بتقليل زمن تنفيذ الاختبار من خلال منع تنفيذه في حال وجود فشل مرحلي. حيث تفيد السياسة المتبعة والموثقة في البحث بتسريع الزمن اللازم لعملية التطوير بشكل عام وفي الأنظمة الكبيرة وبشكل خاص، حيث أنه لا يتم إصدار نسخة لفريق الاختبار إلا بعد التأكد أن الجزء الأساسي من الاختبارات خالية من الأخطاء، وهذا يوفر وقت فريق الاختبار اللازم لتنصيب التطبيق على أجهزتهم والبدء بعملية الاختبار اليدوي واكتشاف هذه الأخطاء.

5- أكّد البحث على النتائج التي تم ذكرها في البحث **[13]** بأن تطبيق الاختبار المستمر يساعد بالتعرف على الأخطاء في مراحل مبكرة من عملية التطوير، كما يمكّن من تنفيذ اختبار كليّ وشامل للتطبيق البرمجي وبعدد غير محدد من المرات، حيث يمكن تنفيذه عند الطلب على عكس الاختبار اليدوي الذي يتطلب موارد بشرية، حيث أن معظم الشركات لا تقوم بتخصيص الموارد اللازمة (أشخاص، موارد مادية، زمن) وهو من الصعب جداً تطبيقه في بيئة الاختبار اليدوي أو حتى في بيئة الاختبار المؤتمت المحلّي.

6- توفر آلية الاختبار المستمرة بيئة معزولة لتطبيق الاختبارات اللازمة، حيث أن المستخدمين ليسوا بحاجة لإعداد بيئات الاختبار على أجهزتهم وهذا مهم جداً خاصة للمستخدمين غير المسؤولين بشكل أساسي عن عملية الاختبار وإنما هدفهم الأساسي هو استعراض نتائج التطبيق النهائي لعملية الاختبار.

7- تميّز البحث عن الأبحاث **[13] [11] [10]** بأننا قمنا بإضافة إعدادات خاصة لتأمين واجهة مناسبة للمستخدمين تمكنهم من اختيار حالات الاختبار اللازمة مع إعداد بيانات الاختبار التي يمكن أن تختلف من مستخدم إلى آخر أو من تطبيق إلى آخر، وهكذا نكون قد عمّمنا عمل

الأداة لتشمل أي خيارات أوسع للمستخدم وابتعدنا عن الخيارات الثابتة والتي تستهلك وقت الاختبار ولا توافق الحاجة دائماً.

10- الاستنتاجات والتوصيات

قمنا في البحث بعرض لمحة سريعة عن منهجية تطوير البرمجيات أجيل والتي تعد الأساس لعمل المؤسسات الكبرى في الوقت الراهن، في الخطوة التالية انتقلنا إلى عرض لمحة عن DevOps وعن التكامل والتسليم المستمر CI/CD. في المرحلة التالية من البحث تمّ طرح المشكلة العلمية المتمثّلة في زيادة في احتمال ظهور الأخطاء نتيجة الإصدارات الكثيرة في حال تمّ اتّباع منهجيّة الأجيل وهذا يتطلب جهود وأوقات إضافية من فريق الاختبار. كما تتمثل المشكلة الأخرى في الكم الكبير من البيانات الناتجة عن الاختبارات والتي لا تعطي صورة واضحة تماماً لأصحاب العمل.

لذلك قمنا بعرض تطبيق عملي لإعداد بيئة اختبار مؤتمت مستمر ضمن خط التكامل والتسليم المستمر على مستويين من مستويات الاختبار وهما اختبار واجهة التطبيق البرمجيّة واختبار واجهة المستخدم. وضّحت النتائج أن استخدام الاختبارات المؤتمتة وتنفيذها ضمن بيئة التطوير المستمر ساهم في كل مما يلي:

1. تقليل الوقت اللازم لتنفيذ الاختبارات المؤتمتة وهذا ما حقق التوازن بين الكلفة والجودة. وبالتالي تسريع عملية إيصال إصدار للتطبيق البرمجي بجودة عالية.
2. تقليل الاعتمادية على تنفيذ الاختبار على جهاز شخص محدد من خلال إيجاد بيئة مشتركة لتنفيذ نوعي الاختبار المؤتمت على مستوى طبقة واجهة التطبيق البرمجية ومستوى واجهة المستخدم.
3. اكتشاف الأخطاء في مراحل مبكرة من منهجية التطوير أجيل، ليقوم فريق التطوير بالعمل على إصلاحها قبل علم فريق الاختبار بها.
4. آلية سهلة للحصول على التغذية الراجعة من خلال توليد التقارير التي تساهم في إعلام أصحاب المصلحة بنتائج الاختبار في مراحل مبكرة وبالتالي اتخاذ القرار في إصدار نسخة من التطبيق أم لا.

يجب أن ننوه أن البحث لم يأتي على ذكر عملية تضمين اختبار الوحدة (Unit Test) ضمن خط تضمين الاختبار لأن هذه الاختبارات يتم كتابتها بشكل أساسي من قبل فريق التطوير ولكن عملية تضمينها ضمن خط التطوير مشابهة تماماً لعملية تضمين اختبارات واجهة التطبيق نظراً لعمومية الخطوات التي قمنا بتنفيذها. كما يجب أن نذكر أنه يمكن على المدى البعيد تصميم بيئة اختبار تتيح للمستخدمين تحديد أكثر من مجموعة اختبار لتنفيذها على التوازي.

المراجع

[1] W. Cunningham, "Principles behind the Agile Manifesto", Agilemanifesto.org, 2017. [Online] Available: http://agilemanifesto.org/principles.html

[2] Haghighatkhah, A., Mäntylä, M., Oivo, M., & Kuvaja, P. (2018). Test prioritization in continuous integration environments. *Journal of Systems and Software*, *146*, 80–98. https://doi.org/10.1016/j.jss.2018.08.061

[3] J. Humble, and D. Farley, "Continuous delivery : reliable software releases through build, test, and deployment automation", 1st ed.Addison-Wesley Professional, 2010

[4] L. Chen, "Continuous Delivery: Huge Benefits, but Challenges Too", IEEE Software, vol. 32, no. 2, pp. 50-54, 2015A

[5] Kandil, P., Moussa, S., & Badr, N. (2014). Regression testing approach for large-scale systems. IEEE International Symposium on Software Reliability Engineering Workshops, 2014, 132–133.

[6] H. Liu, Z. Li, J. Zhu, H. Tan, H. Huang "A Unified test framework for continuous integration testing of SOA solutions " International Conference on Web Services 2009

[7] S. Elbaum, G. Rothermel, J. Peni "Techniques for improving regression testing in continuous integration development environments" International Symposium on Foundations of Software Engineering 2014

[8] J. Lu, Z. Yang, J. Qian "Implementation of continuous integration and automated testing in software development of smart grid scheduling support system" International Conference on Power System Technology 2014

[9] M. Brandtner, E. Giger, H. Gall " SQA-Mashup: A mashup framework for continuous integration" Information and Software Technology 2015

[10] M.Callanan and A. Spillane, "DevOps: Making It Easy to Do the Right Thing", in IEEE Software. 33. 1-1. 10.1109/MS.2016.66, 2016

[11] J. Gmeiner, R. Ramler, J. Haslinger "Automated testing in the continuous delivery pipeline: A case study of an online company" User Symposium on Software Quality, Test and Innovation 2015

[12] T. Karvonen, L.E. Lwakatare, T. Sauvola, J. Bosch, H.H. Olsson, P. Kuvaja, M. Oivo "Hitting the target: Practices for moving toward innovation experiment systems" International Conference on Software Business 2015

[13] Soni, Mitesh. "End to end automation on cloud with build pipeline: the case for DevOps in insurance industry, continuous integration, continuous testing, and continuous delivery." 2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM). IEEE, 2015.

[14] .A.I.B.S. Arachchi, Indika Perera "Continuous Integration and Continuous Delivery Pipeline Automation for Agile Software Project Management " Moratuwa Engineering Research Conference (MERCon) 2018

[15] E. Collins, A. Neto, and V. Lucena Jr, "Strategies for Agile Software Testing Automation: An Industrial Experience". 440-445. COMPSACW.2012.84, 2012

[16] T. Savor, M. Douglas, M. Gentili, L. Williams, K. Beck, and M. Stumm, "Continuous deployment at facebook and oanda," In Proc. 38th International Conference on Software Engineering Companion, 2016. pp. 21–30

[17] Wieringa, R., Maiden, N., Mead, N., & Rolland, C (2006). Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. Requirements Engineering, Vol. 11,No. 1, pp. 102–107. DOI: 10.1007/s00766-005-0021-6.

[18] K. Alhamazani, R. Ranjan, K. Mitra, F. Rabhi, P. Jayaraman, S.Khan, A. Guabtni, and V. Bhatnagar, "An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the- art", in Computing, vol. 97, no. 4, pp. 357-377, 2014

[19] S. Jarkas, E. Esper, N. Abu Saleh. "Applying Automated Testing of Event-Driven Software Systems" In Al-Baath university journal vol. 42 No. 21, 2020

[20] S. Jarkas, E. Esper, N. Abu Saleh "DESIGN AND IMPLEMENT AN AUTOMATED RESTFUL API TESTING FRAMEWORK" «Научно-практический электронный журнал Аллея Науки» №12(51) 2020 Alley-science.ru