

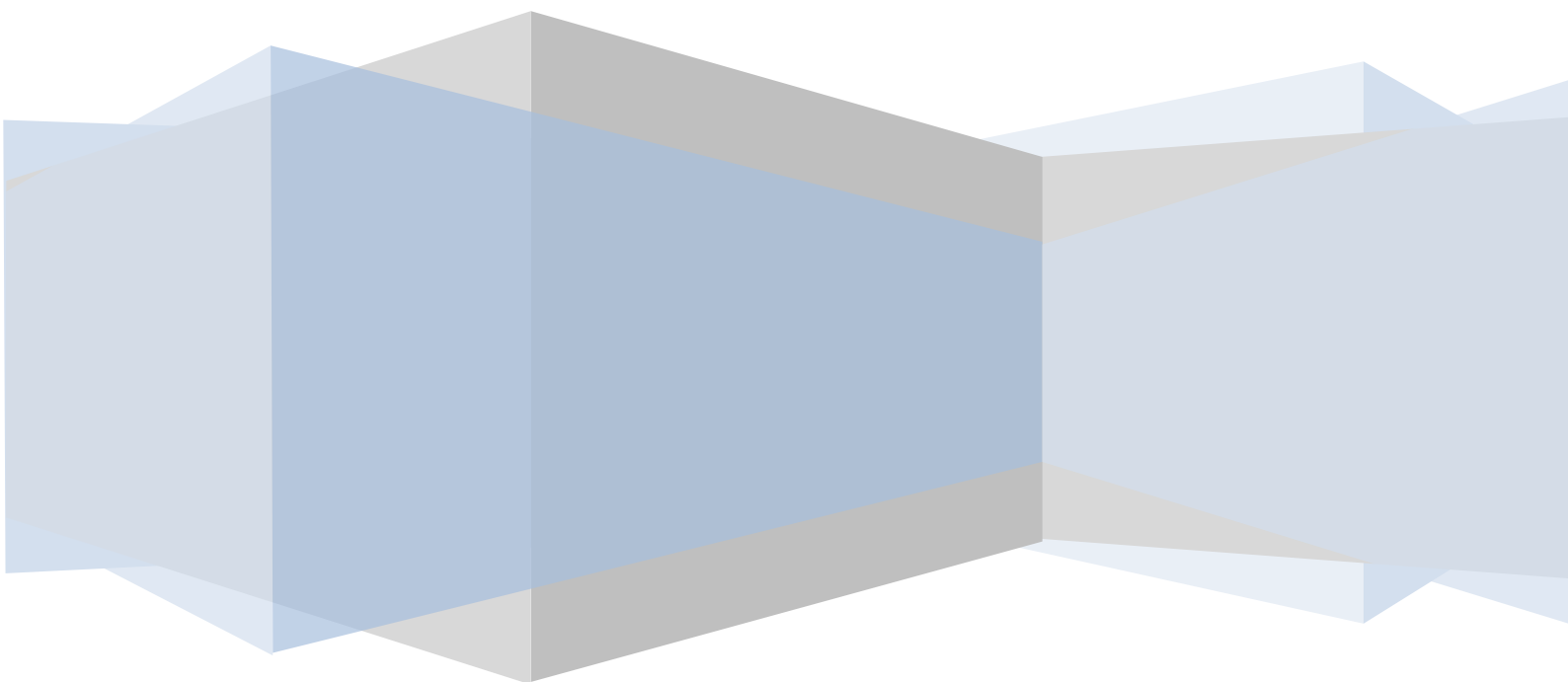


Projet xCluster

Interface Web pour la Classification Croisée

Ted RAMONI - Quentin CHAUVEL - Valentin LESIEUR - Selma CAGLAYAN

Encadrant : Aghiles SALAH - François ROLE



Sommaire

I.	Contexte du sujet	3
II.	Objectifs du projet	3
III.	Organisation du projet	4
IV.	Lotissement du projet	4
V.	Environnement technique	6
VI.	Architecture.....	8
VII.	Scénario d'utilisation.....	10
VIII.	Axes d'amélioration	18
IX.	Table des illustrations	19
	Annexe.....	20

I. Contexte du sujet

Avec l'expansion des volumes de données disponibles, notamment sur le web, la classification ne cesse de gagner en importance dans le domaine de la science des données pour la réalisation de différentes tâches, telles que le résumé automatique, l'accélération des moteurs de recherche, l'organisation d'énormes ensembles de données, etc. Dans ce contexte, les méthodes de classification croisée, consistent à **partitionner simultanément les lignes et les colonnes d'une matrice**, s'avèrent être plus avantageuses que les méthodes de classification simple pour plusieurs raisons telle-que la rapidité et l'interprétation des résultats.

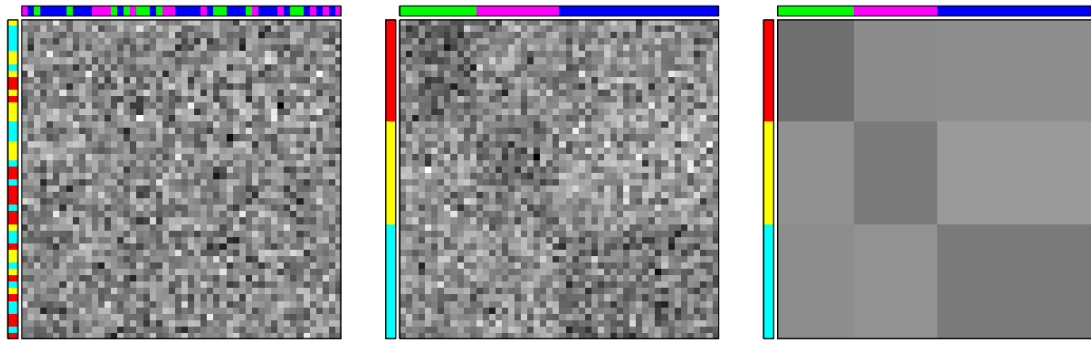


Figure 1 - Représentations graphiques d'une matrice avant et après traitement

II. Objectifs du projet

Dans ce projet, on s'intéresse au package python Coclust qui implémente plusieurs méthodes de classification croisée. L'objectif est de développer une interface web pour Coclust afin de le rendre plus accessible et plus simple à utiliser. En particulier cette interface devrait permettre :

- La gestion (création, modification et suppression) d'un répertoire de travail.
- L'importation de jeux de données sous différents format (.mat, .csv, .txt, .xls, etc.).
- L'exécution d'une classification croisée sur un jeu de données et récupération des résultats.
- La visualisation et la sauvegarde des résultats.

Par exemple, l'utilisateur crée un répertoire de travail, il importe ses données, il applique une classification croisée sur ses données en indiquant la méthode de son choix, le nombre de classes lignes et colonnes, les paramètres spécifiques à l'algorithme choisi.

III. Organisation du projet

Le projet a été planifié à l'aide d'un diagramme de GANTT avec une vision macroscopique (vision globale avec moins de précision) des tâches du projet pour gérer le rythme non régulier des séances et les évolutions dans les fonctionnalités au fur et à mesure du projet.

Le diagramme de GANTT est réalisé sur « Google Sheets » et partagé avec l'ensemble de l'équipe et des encadrants pour que tout le monde est une visibilité sur l'avancement du projet.

Le suivi du projet a été effectué avec l'encadrant en méthode agile avec à minima une réunion par semaine pour faire un point d'avancement sur le développement et voir les éventuelles problématiques.

Dans le cadre des développements, la répartition des tâches a été effectué par un système en Kanban (système de tickets) à l'aide du service fourni par GitHub. Cela permet à chaque développeur de s'attribuer une tâche qui va ainsi suivre le processus suivant :

- En cours
- Fait
- Supprimé

Les tâches à effectuer sont réunis dans une colonne « TODO », lorsqu'un développeur s'affecte une tâche, celle-ci passe dans la colonne en cours et une fois que cette tâche est développée puis testé alors elle passe dans la colonne « Fait ».

Les tâches abandonnées par le client sont stockées dans la colonne « Supprimé » pour une éventuelle reprise future.

IV. Lotissement du projet

Le projet a été réparti en 6 lots permettant d'avoir des lots fonctionnels et testables pour rapidement détecter des problèmes d'environnement ou de fonctionnalité.

Nous avons donc dans un premier temps décidé de faire un lot sous un format PoC (Proof of Concept) de communication entre NodeJS et Python pour ensuite faire un lot avec l'implémentation d'une méthode de la librairie coclust.

Une fois une méthode à minima implémentée nous avons décidé de faire un lot pour les gestions des utilisations.

À ce niveau il ne reste plus qu'à implémenter les méthodes restantes de la librairie coclust pour ensuite finir sur le design et la mise en production.

Voici ci-dessous le détail de chaque lot :

LOT 1 : Installation de l'environnement

Installation de l'environnement applicatif (serveurs node, angular, python ...) et leur interconnexion.

LOT 2 : Implémentation CoClustMod

Implémentation de la fonction CoClustMod dans l'application (sélection de la fonction, saisie des paramètres et affichage des résultats (textes, représentation graphique)).

LOT 3 : Authentification / Gestion Workspace

Gestion des utilisateurs avec inscription, connexion, création du dossier utilisateur, des workspaces et restriction des accès.

LOT 4 : Implémentation des autres fonctionnalités

Implémentation des autres fonctions de la librairie CoClust.

LOT 5 : Design & Branding

Modification du design de l'application et branding de celle-ci (CSS, image, logo ...).

LOT 6 : Mise en production

Mise en production de l'application sur un serveur.

V. Environnement technique




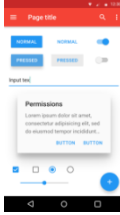
Front-end	
	<p>NodeJS est une plate-forme de programmation JavaScript orientée vers les applications réseau qui doivent pouvoir monter en charge.</p>
 ANGULARJS	<p>AngularJS est un framework JavaScript libre et open-source développé par Google.</p>
 Bootstrap	<p>Bootstrap est une collection d'outils utile à la création du design (graphisme, animation et interactions avec la page dans le navigateur ... etc. ...) de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option. C'est l'un des projets les plus populaires sur la plate-forme de gestion de développement GitHub.</p>
	<p>Material Design est un ensemble de règles de design proposées par Google et qui s'appliquent à l'interface graphique des logiciels et applications. Il est utilisé notamment à partir de la version 5.0 du système d'exploitation Android.</p>

Figure 2 - Tableau des technos utilisées pour le Front-End

Back-end



Python est un langage de programmation objet, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est ainsi similaire à Perl, Ruby, Scheme, Smalltalk et Tcl.



NodeJS est une plate-forme de programmation JavaScript orientée vers les applications réseau qui doivent pouvoir monter en charge.



MongoDB est un système de gestion de base de données orientée documents, répartissable sur un nombre quelconque d'ordinateurs et ne nécessitant pas de schéma prédéfini des données.



PHP est un langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. PHP est un langage impératif orienté objet.

Figure 3 - Tableau des technos utilisées pour le Back-End

VI. Architecture

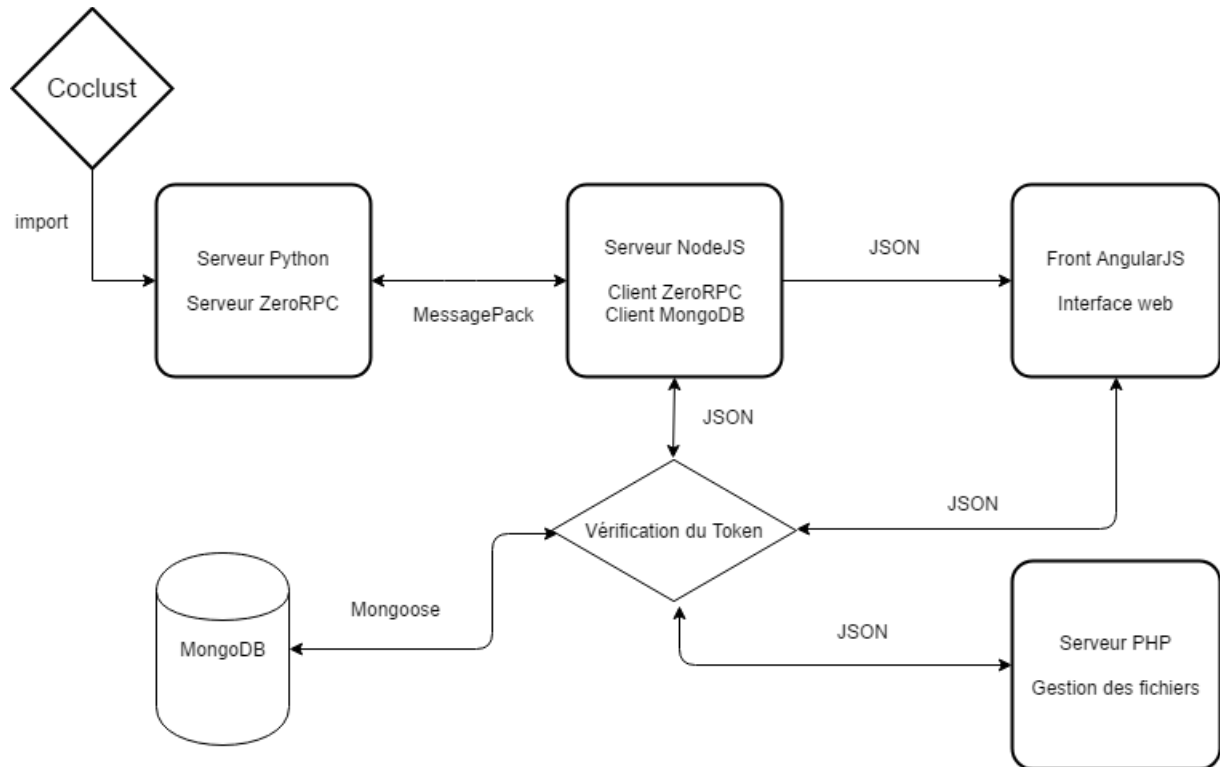


Figure 4: Architecture xCluster

Nous avons 5 serveurs :

- **Un serveur python** (serveur zeroRPC), api attendant des requêtes, traite les requêtes et retourne les résultats en utilisant la library coclust.
- **Un serveur NodeJS** (client zeroRPC), api attendant des requêtes du front et fait le pont entre le serveur python et le front.
- **Un serveur AngularJS** (front) utilise le module file-manager pour la création du workspace utilisateur, upload des fichiers, etc...
- **Un serveur MongoDB** (base de données) qui contient les identifiants des utilisateurs de xCluster.
- **Un serveur PHP** pour la gestion des fichiers et documents. Il permet par exemple de renommer les fichiers et répertoires ou de les déplacer.

Ces serveurs communiquent entre eux via différentes technologies. Ainsi la communication entre notre back-end NodeJS et l'API Python de coclust se fait avec zeroRPC qui est une implémentation de zeroMQ. Cette technologie est basée sur les web sockets. L'utilisation de cette technologie nécessite l'installation et l'utilisation d'une bibliothèque Python et d'une autre NodeJS toutes deux configurées pour écouter et diffuser sur le même port.


Le reste des communications entre les différents serveurs se font via des appels REST (Representational State Transfer) sur le protocole http (HyperText Transfer Protocol). Ainsi lorsque l'utilisateur lance une classification croisée sur l'interface web, le front-end AngularJS envoie une requête REST au back-end NodeJS qui vérifie le token de l'utilisateur en communiquant avec la base de données MongoDB via Mongoose. Par la suite NodeJS envoie une requête à l'API coclust via zeroRPC qui renvoie le résultat via cette même technologie et qui est de nouveau renvoyé à AngularJS en réponse à la requête REST.

Pour les actions nécessitant de passer par le serveur PHP, comme un déplacement de fichier, la requête REST est envoyée au serveur PHP qui lui-même envoie une requête REST à NodeJS via cURL pour vérifier l'accès à l'action avec le token de l'utilisateur comme expliqué précédemment. Si l'accès est autorisé le serveur PHP effectue l'action et renvoie un résultat en réponse à la requête REST envoyée par AngularJS.

VII. Scénario d'utilisation

Ci-dessous, un scénario nominal d'utilisation de xCluster, chaque point décrit ci-dessous correspondront dans l'ordre aux captures d'écrans suivantes :

- I. Connexion ou ...
- II. ... inscription de l'utilisateur.
- III. Accès à son workspace (creation de dossiers, navigation, upload de fichiers).
- IV. Ouverture du menu contextuel d'actions sur un fichier. mat, .csv ou autre et sélection d'un traitement de classification croisée.
- V. Remplissage du formulaire pour définir les paramètres
- VI. Présentation des résultats et génération des fichiers contenant les informations
- VII. Téléchargement ou prévisualisation des résultats.



Identifiant


Mot de passe

CONNEXION

Pas encore inscrit? [Créer un compte](#)

Accès à la fenêtre d'inscription

Figure 5 - I. Connexion



The logo consists of a blue hexagon with a 3D effect, composed of three overlapping planes. Below the hexagon, the word "XCLUSTER" is written in a bold, blue, sans-serif font.

Identifiant

Mot de passe

Adresse email

CRÉER LE COMPTE

Déjà inscrit? [Se connecter](#)

Figure 6 - II. Inscription

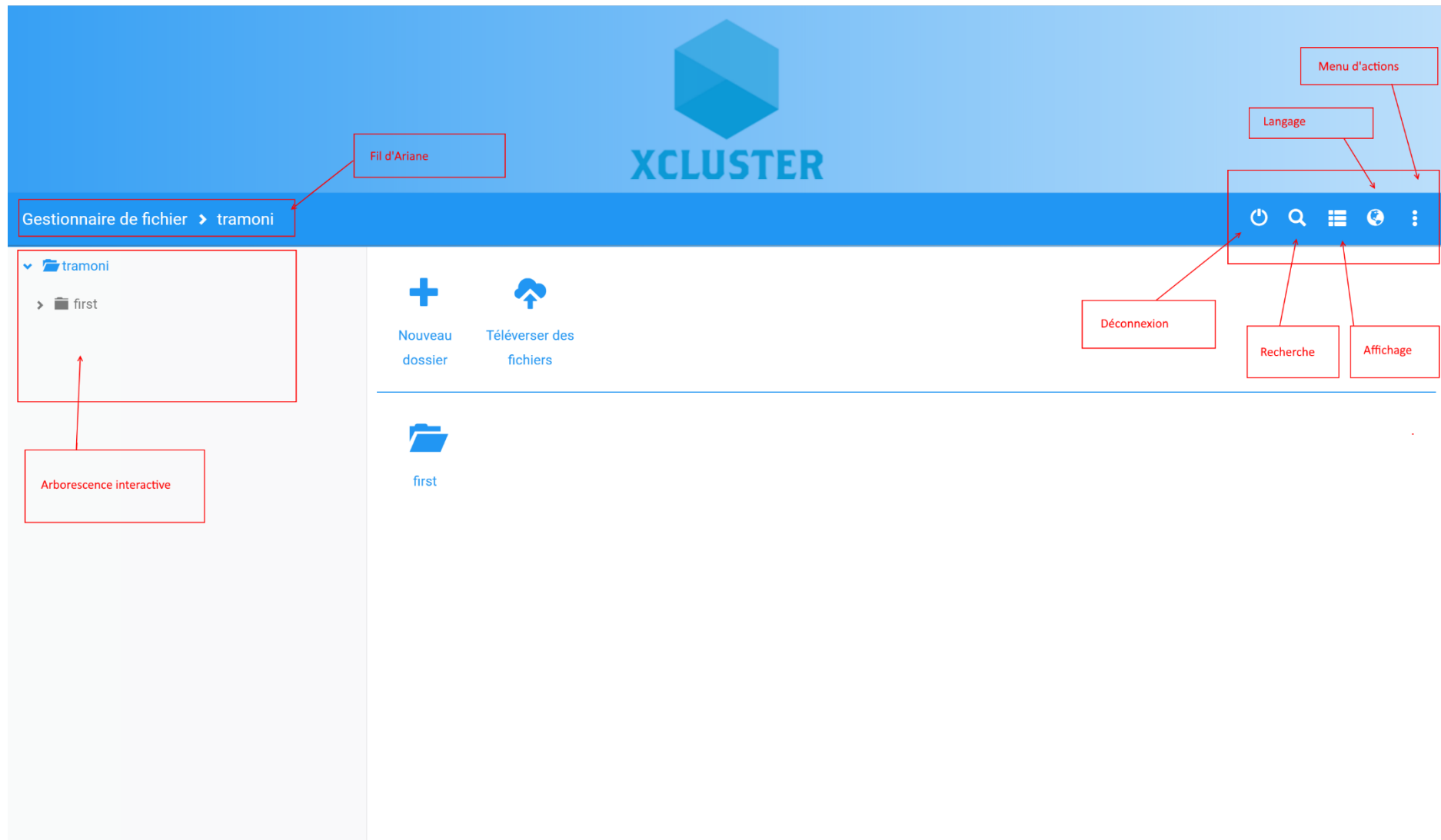


Figure 7 - III. Workspace

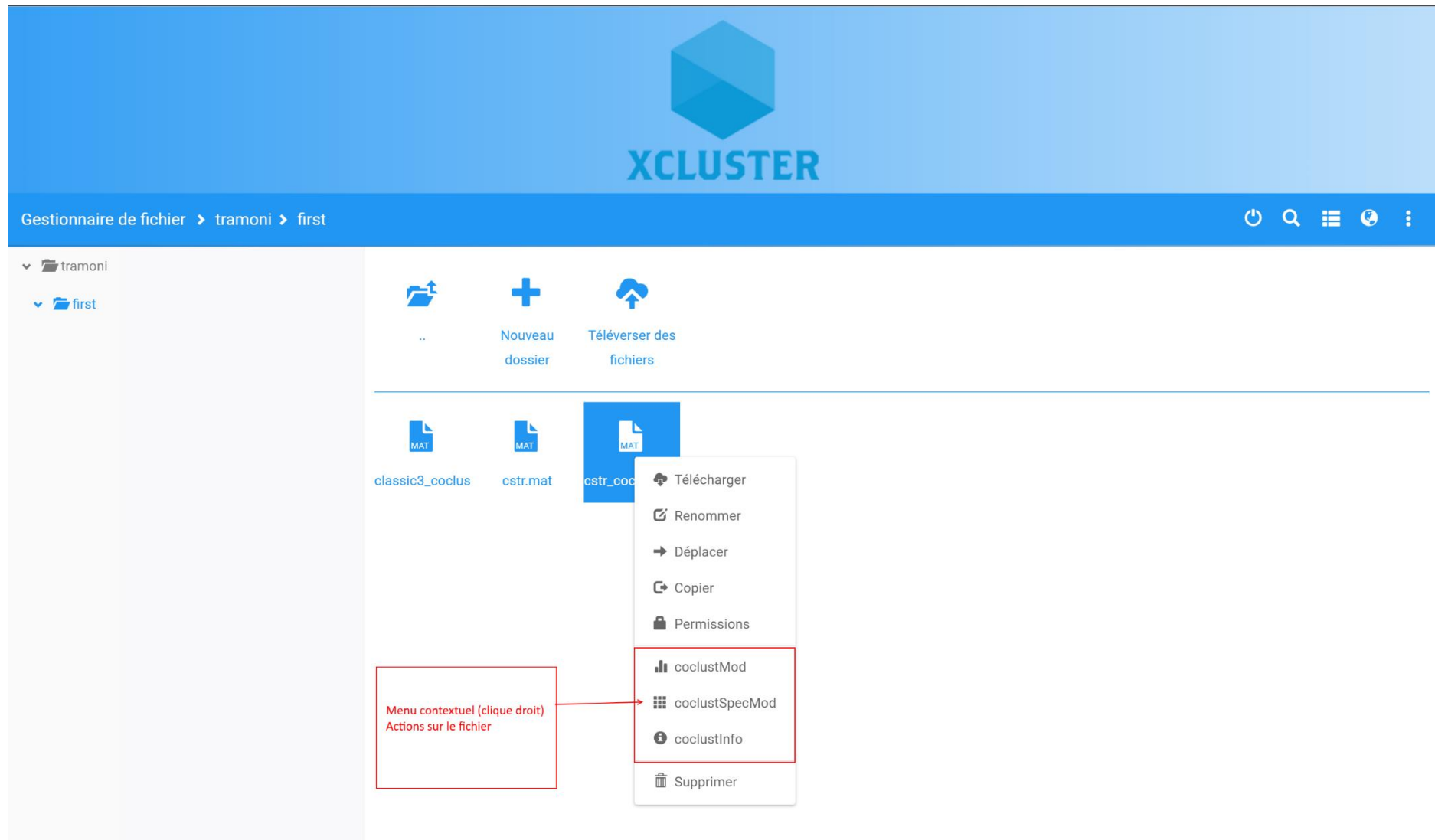


Figure 8 - IV. Menu contextuel

Gestionnaire de fichier > tramon > first

tramon

first

Si fichier de type coclust format il est possible de récupérer les top termes.

Si fichier de type coclust format il est possible de récupérer les top termes.

CoclustMod

Nombre de clusters (n_clusters): 2

Label de colonne initiale (init):

Maximum d'itérations (max_iter): 20

Nombre d'initialisations (n_init): 1

Etat aléatoire (random_state):

Tolérance relative (tol): 1e-9

Nom de la matrice (mat): doc_term_matrix

Récupérer les top termes ☐

ANNULER CONFIRMER

Figure 9 - V. Formulaire des paramètres

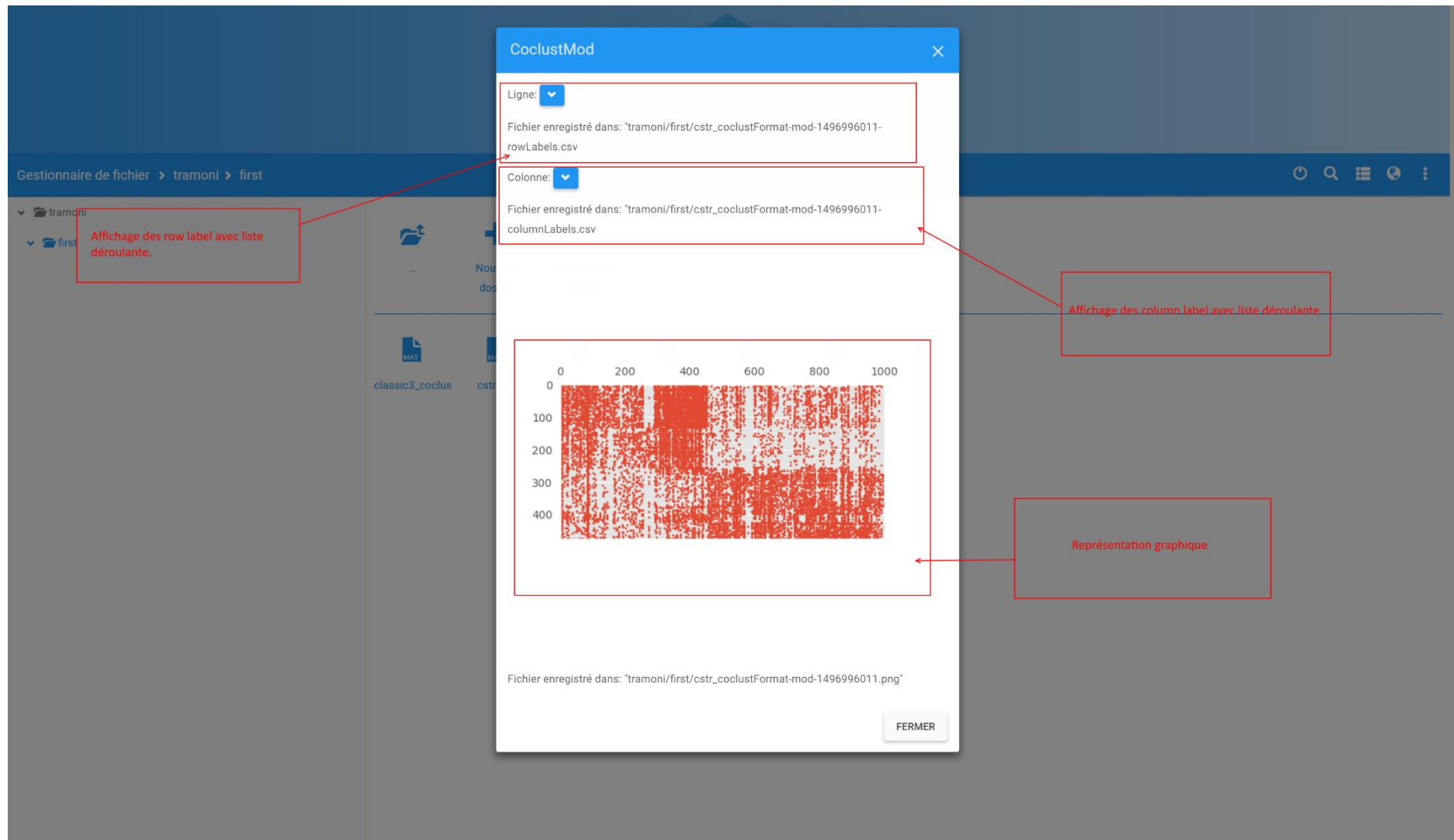


Figure 10 - VI. Résultats

XCLUSTER

Gestionnaire de fichier > tramoni > first

Nom ▼	Taille	Date
...		
+ Nouveau dossier Téléverser des fichiers		
classic3_coclustFormat.mat	2.1 MB	2017-05-11 16:51:26
cstr.mat	75.2 kB	2017-05-11 15:56:58
cstr_coclustFormat-mod-1496996011-columnLabels.csv	25.4 kB	2017-06-09 10:13:31
cstr_coclustFormat-mod-1496996011-rowLabels.csv	12.1 kB	2017-06-09 10:13:31
cstr_coclustFormat-mod-1496996011.png	84.4 kB	2017-06-09 10:13:31
cstr_coclustFormat.mat	194.0 kB	2017-05-11 15:54:47

Fichiers générés lors du traitement téléchargeables (ou prévisualisables dans le cas de fichiers au format png, jpeg, svg, etc.)

Figure 11 - VII. Fichiers générés

VIII. Axes d'amélioration

Le code comporte actuellement des duplications et manque de services pour mieux les répartir. Il nécessite donc un refactor technique.

L'application est composée d'un trop grand nombre de serveurs. Le serveur PHP pourrait notamment être remplacé par du code Python. Cela permettrait de réduire le nombre de serveur mais aussi de rendre l'application plus maintenable.

L'application ne comporte actuellement aucun test. Des tests unitaires voire fonctionnels seraient les bienvenus toujours dans le sens d'une amélioration de la maintenabilité et la prévention des régressions.

Enfin de nouvelles fonctionnalités pourraient être ajoutées. En effet, de nouveaux algorithmes sont prévus pour l'API coclust, ceux-ci pourraient aussi être ajoutés sur l'application xCluster. Les graphiques pourraient aussi devenir interactifs pour une meilleure expérience utilisateur et une meilleure visibilité de l'information. Une fonction de vérification antivirus des fichiers téléversés serait aussi la bienvenue pour la sécurité. Un guide d'utilisation interactif pourrait aussi être ajouté.

IX. Table des illustrations

Figure 1 - Représentations graphiques d'une matrice avant et après traitement	3
Figure 2 - Tableau des technos utilisées pour le Front-End	6
Figure 3 - Tableau des technos utilisées pour le Back-End	7
Figure 4: Architecture xCluster.....	8
Figure 5 - I. Connexion.....	11
Figure 6 - II. Inscription.....	12
Figure 7 - III. Workspace.....	13
Figure 8 - IV. Menu contextuel.....	14
Figure 9 - V. Formulaire des paramètres.....	15
Figure 10 - VI. Résultats.....	16
Figure 11 - VII. Fichiers générés	17

Annexe

Planification du projet

https://docs.google.com/spreadsheets/d/1geq_MT5jTGGbG8MeCRF-Ym3qbRrRIxLKY1DfkgS3DCg/edit?usp=sharing

Kanban du projet

<https://github.com/groschatchauve/xCluster/projects/1>

Installation de l'environnement serveur pour xCluster

Ci-dessous nous détaillons comment installer l'environnement pour faire tourner xCluster sur un serveur.

Dans cette procédure nous utilisons une machine sous Windows 10.

L'application est constituée de 2 parties :

- Une partie back en python qui fonctionne en tant que serveur et attend des requêtes du client pour l'exécution des fonctionnalités du package CoClust
- Une partie front en node.js qui est le client du serveur python et enverra les requêtes de l'utilisateur par le biais de l'interface graphique et affichera ensuite les résultats sur cette même interface à la réception du traitement du serveur.

Installation de ZeroMQ

Télécharger la dernière version stable de *ZeroMQ* sur leur site internet :

<http://zeromq.org/distro:microsoft-windows>

Note :

- Dans le cadre d'une production sur un serveur Azure, il peut être difficile d'installer ZeroMQ sans un accès direct/distant à la machine pour l'exécution de l'installateur.
- Dans notre cas nous travaillons en local sur notre machine.

Installation des composants pour le serveur Python

Prérequis :

- Installation de Python version 2.7, nous conseillons de passer par la distribution Anaconda pour avoir tous les composants requis au bon fonctionnement de l'application et de CoClust. La distribution 2.7 de Python est disponible à l'adresse suivante sous la version 4.3 d'Anaconda: <https://www.continuum.io/downloads>

Nous allons maintenant installer les modules node.js requis au déploiement de l'application

Dans un premier temps la librairie python coclust doit être installée pour cela veuillez vous référer à la documentation d'installation ci-dessous :

<http://coclust.readthedocs.io/en/v0.2.0/install.html>

Après l'installation de coclust, il faut installer les packages pour l'utilisation de zeroRPC :

- `pip install zerorpc`
- `pip install msgpack-python --force-reinstall --upgrade`

Installation des composants pour le client Node.js**Prérequis :**

- Dans un premier nous conseillons activement d'avoir les librairies C++ Visual Studio 2015 et les outils de compilation associés. Pour cela Microsoft propose un standalone à installer répondant aux prérequis ci-dessus :
<http://landinghub.visualstudio.com/visual-cpp-build-tools> OU exécuter ces commandes :
- `npm install --global --production windows-build-tools`
- `npm config set msvs_version 2015`
- Il faut ensuite avoir Node.js installer sur sa machine pour ce faire nous conseillons de télécharger la dernière version stable de Node.js disponible sur leur site internet :
<https://nodejs.org/en/>

Nous allons maintenant installer les modules node.js requis au déploiement de l'application

- `npm -g install npm@next`
- `npm install -g bower`
- `npm install node-gyp`
- `npm install zerorpc`
- `npm install zmq`

Installation de xCluster**Prérequis :**

- Installation de l'environnement serveur

- Récupération du projet sur Git (git pull)

Le lancement et l'arrêt de xCluster se fait par le biais de fichier automatisant ces processus :

start.vbs : Permet de lancer tous les serveurs au bon fonctionnement de l'application, au premier démarrage, il sera demandé de fournir le chemin vers le binaire php.exe. A la fin du processus, l'application se lance sur le navigateur.

stop.bat : Permet d'arrêter tous les serveurs.

Utilisation de xCluster

Création d'un compte, connexion et déconnexion

Dans la page d'accueil de xCluster, une boîte de connexion au centre de la page permet de se connecter à l'application par la saisie de son identifiant (login) et de son mot de passe, le succès de la validation des informations saisies redirige l'utilisateur vers son workspace.

Dans la boîte de connexion, un bouton en bas à droite, un lien "Créer un compte" permet d'afficher la boîte d'inscription. La validation des informations saisies connecte directement l'utilisateur vers son workspace.

Sur le workspace, un bouton de déconnexion, en haut à droite de l'écran, permet de se déconnecter.

Création d'un dossier dans le workspace

Il existe plusieurs méthodes pour créer un dossier dans le workspace :

- A l'aide de la barre de navigation
- A l'aide de l'icône dans le dossier courant
- A l'aide du menu contextuel

Téléversement d'un fichier

Il existe plusieurs méthodes pour créer un fichier dans un dossier :

- A l'aide de la barre de navigation
- A l'aide de l'icône dans le dossier courant
- A l'aide du menu contextuel
- Par le biais d'un drag & drop dans la zone d'affichage des fichiers

Modification d'un fichier ou dossier

Pour chaque dossier / fichier, un clique droit permet d'ouvrir un menu contextuel et permet de renommer, modifier un fichier.

Suppression d'un fichier ou dossier

Pour chaque dossier / fichier, un clic droit permet d'ouvrir un menu contextuel et permet de supprimer un fichier.

Export des résultats

Lors du traitement des fichiers, les données générées (graphiques, excel) sont automatiquement exportées dans des fichiers dans le dossier courant. Ces fichiers respectent la politique de nommage suivante : **nom_du_fichier**-info-nombre-typeDeDonnées.FormatFichier

Chaque fichier peut ensuite être télécharger à l'aide de l'option dans le menu contextuel.

Installation de MongoDB

Télécharger la version community 3.4 à partir de ce lien :

https://www.mongodb.com/download-center?jmp=docs&_ga=1.266151433.748348421.1492676042#community
