Nono-Facture
Entités Nommées

par Algonano

Yacine BOUKHELIF
Daphné HERAIZ-BEKKIS

30 Juin 2016

Attentes - côté utilisateur

Entrée :

Facture au format PDF

Sortie :

Fichier Excel

Type : Application logicielle à télécharger

Ouverture (Ctrl+O) ou DragAndDrop du fichier facture dans le logiciel
Ouverture automatique (ou avec autorisation) du retour dans Excel
Paramètres modifiables pour l' utilisateur
Ouverture possible depuis n'importe quel dossier
Sauvegarde possible dans n'importe quel dossier

Disponible pour Linux, Windows et Mac

Attentes - côté entreprise

Info sur l'expéditeur de la facture	Nom, type (SARL…)+capital de l'entreprise, RCS (ville et numero), Adresse envoyeur, Tel+Fax, Num TVA (FR)
Info sur le destinataire de la facture	Nom, type (SARL…),à l'attention de?,Adresse receveur,Tel+Fax,Num TVA (FR)
Facture	Titre ?,Numero, date
Infos factures simplifiées	Bloc de texte prestations, montant total HT, taux TVA, mont total TVA, montant total autres taxes, montant total TTC, moyen de paiement ?
Infos factures (groupement de toutes les prestations)	montant HT, taux TVA, montant TVA, autres taxes, montant TTC(total), moyen de paiement
Infos pour une prestation	Nature prestation,montant HT, taux TVA, montant TVA, autres taxes, montant TTC(total prestation)

Utilisation

PDF → TXT
OCR
Retour d'une OCR : Txt /html/xml

Séparation des factures ?

Lecture et classification des données

Ecriture Sortie → Fichier CSV (fichier texte avec séparation ;)
CSV lisible partout et réadaptable par excel pour une lecture classeur
Conversion Excel

Ajout : Possibilité pour l'utilisateur de préciser la présence d'une erreur ce qui permettra au système d'ajouter un entraînement supplémentaire

Ce qui a été fait ...

- CSV -> Excel (avec gestion des versions) avec openpyxl,xlrd,xlwt
- PDF -> TXT/HTML/XML
- Structure de facture
- Regexp (descripteurs)

Etapes de conception

1. Tout au long étude et réflexions sur les divers algorithmes existants
2. Création fichier Excel en python + lecture (Excel ↔ CSV)
3. Lecture et retour OCR à partir de PDF (+ PDF ↔ PNG)
4. Détermination d'une structure de facture
5. Classification par regexp
6. Classement manuel des factures dans la base
7. Codage de l'algorithme (tests et comparaisons de multiples algo – à expliciter)
8. Apprentissage avec la basse de données (validation croisée ?)
9. Binding du tout
10. Ecriture de la documentation
11. Création de l'exécutable + installation automatique de toutes les librairies nécessaires au bon fonctionnement de l'appli
12. Gestion de toutes les erreurs possibles
13. Tests de validation de toutes les options et paramètres possibles
14. Tests par un tiers
15. Création IHM (+initiation au drag&drop)

Ce qu’il reste à faire ...

- Mise en place base apprentissage : Classification manuelle des factures
- Codage Algorithme
- Interface graphique
- Gestion des problèmes . . .

Ce qui pose problème ...

- Lecture de Scanned PDF
- Drag&Drop interface
- Encoding des textes

Extraction des données - PDF to HTML

- ➔ PDF to HTML : OCR avec pdfminer
- ➔ Choix HTML car :
  - ➢ Position absolue des éléments
  - ➢ Séparation par blocs
- ➔ Gestion des PDF scannés
  - ➢ Scanned PDF to PNG
  - ➢ PNG to PDF + PDF to HTML
  - ➢ PNG to HTML

Retour des professeurs

Retour Vincent CNAM

- ➔ pourquoi insiter sur l'apprentissage ?
- ➔ générer aléatoirement une plus grosse base de données de factures
- ➔ faire de l'apprentissage sur les clusters/bloc pour identifier le destinataire et l'expéditeur et les données factures et les données prestations

Retour Vincent Guigue LIP6

- ➔ système AdHoc de reconnaissance (par ex les factures free gardent la même architecture etc.)
- ➔ peu d'apprentissage
- ➔ vecteurs features par blocs basés sur un maximum de données (taille, nombre de ligne, taille bloc, position, 0 ou 1 pour chaque descripteur regexp possible…)
- ➔ HMM possible mais très probablement inutile
- ➔ ajouter un choix par élimination (si une info a déjà été determiné comme le rcs ou la tva on n'a plus besoin de les tester)

Différenciation des blocs

- Pourquoi?
- Comment?
- Vecteur de classification :

  ➔ (valuationBOW , position, nb de lignes, taille bloc, reg_flags)

Classification intra-bloc

- Reg_flags (TYPE) : tél, adresse, RCS, numéro TVA, montant, type société . . .
- Système AdHoc
- Par élimination

Apparence

- Drag and drop de facture PDF
- Ouverture automatique du fichier généré
- Interface intuitive
- QtCreator -> PyQt4

WebService Crawler

pour Innopolis

Organisation

Identification des informations à crawler (H20H20 fait, Cosme à faire avec Cédric)

Mise en place du crawler (difficulté pages gérénés dynamiquement avec du JS)

Mise en place d’un client pour la base de donnée (réussi en local , reste à faire pour AWS amazon web service, tache assez difficile)

Déploiement du crawler sur le serveur et automatisation de celui-ci.

Création de l’interface utilisateur pour pouvoir afficher les données récuperés par le crawler

Site Web Algonano Déploiement

pour Algonano

Difficultés rencontrées

Plateforme de déploiement : AWS (Amazon)

Opérateur DNS : Gandi.net

- ➢ Relier le DNS sur gandi à l'instance sur Amazon :
  Enregistrement A
- ➢ Accéder immédiatement à un sous-dossier de l'adresse IP référençant l'instance
  add VirtualHost
- ➢ Que l'accès par URL à <DNS>/<qqch> redirige bien
  .htacces add Overwrite

- ➔ Perte du CSS
- ➔ InternalServerError