FOSDEM 2024

Conférence 1 : "A murder party with Lea"

Projet présenté par *Pierre Denis*, qui travaille comme ingénieur logiciel à *Space*.

Présentation générale Lea (qui se prononce "lia" - L'IA) est un programme Python créé pour aider à **calculer des probabilités**, intuitif et facile à utiliser. Ses capacités sont présentées sur la base d'un scénario de résolution d'enquête policière :

Le Dr Black a été trouvé mort. Quatre suspects potentiels devront être étudiés. Nous disposons, pour chacun, de la probabilité qu'il soit coupable : - Colonel Mustard - 40% - Mrs Peacock - 25% - Mrs White - 10% - Professeur Plum - 25%

Nous possédons également des informations fournies par Sigmund, un profiler *de génie : - Si Mrs White est l'assassin, alors il y a une probabilité de 95% qu'elle ne se présente pas à l'interrogatoire, contre 20% sinon. - Si Mrs Peacock est innocente, alors il y a une probabilité de 75% qu'elle sache qui est l'assassin. - Si Mrs Peacock est l'assassin ou si elle sait qui est l'assassin, alors il y a 50% de chance pour qu'elle soit ivre. Si elle est innocente, cette probabilité tombe à 10%. - Si le Colonel Mustard est l'assassin, alors il y a 75% de chance que le Professeur Plum l'accuse, et 5% sinon.*

Ces informations peuvent ensuite être transcrites dans le **langage Python** pour être exploitées par Lea.

Les fonctionnalités de Lea Lea fonctionne sur le modèle d'un **lancer de dé** ou d'un jeu de **pile ou face** : on programme ce dé en fonction des probabilités que l'on veut traiter. Par exemple, un dé peut tirer au sort le coupable selon les probabilités initiales : 1/10 pour Mrs White, 1/4 pour Mrs Peacock et le Professeur Plum, 4/10 pour le Colonel Mustard. La bibliothèque Matplotlib est intégré au programme, ce qui permet de voir les résultats sous forme d'histogrammes et, en cas de grand nombre de lancers, la fonction Lea.val() affiche les résultats obtenus.

Pour les problèmes plus complexes, comme celui de cette enquête policière, Lea permet aussi de manipuler les probabilités conditionnelles, et donc de définir des Booléens afin de pouvoir les mettre en place. Cela se fait en combinant les différents dés.

Plusieurs problèmes de probabilités classiques sont ainsi facilement traitées par Lea : un exemple issu de *The Doctrine of Chance* d'Abraham de Moivre (calculer la probabilité de tomber sur l'as avec trois lancés) ; le célèbre paradoxe des deux enfants ; le problème de Monty Hall.

Lea sera également utile aux joueurs de RPG, s'il faut calculer ses chances de gagner un combat contre un ennemi. Associée à une liste de mots et de règles

de grammaire, Lea est aussi un très bon générateur de phrases aléatoires.

Probabilités et affaires judiciaires Le traitement du scénario présenté ci-dessus se fait donc grâce à la combinaison des informations fournies. On construit un réseau bayésien pour déterminer les probabilité général puis, à partir d'une liste de preuves, on pose les conditions qui mènent au résultat final (par exemple : *Mrs White n'est pas venue à l'interrogatoire*, *Mrs Peacock est ivre*, *Le tueur est une femme*, ...)

Si la présentation s'appuie sur un scénario ludique, une *murder party*, Pierre Denis rappelle le rôle que peut avoir le calcul de probabilités dans les affaires judiciaires réelles. En particulier, il rappelle le cas du bordereau de l'Affaire Dreyfus, qui s'est fondé sur un mauvais calcul de probabilités.

Conférence 2 : "Building the world's virtual classroom"

Cette conférence, initialement proposée par Fred Dixon et Steven Muegge, n'a été présentée que par ce dernier, son collègue étant absent. Steve Muegge est le directeur exécutif de "BigBlueButton", la plateforme dont il est question ici.

"BigBlueButton", une introduction *BigBlueButton* est une plateforme de visioconférences créée en 2007 à l'Université de Carlton, utilisée pour construire des classes virtuelles, fondé sur un LMS (learning management system) : son objectif est de développer l'éducation à distance. Les cours en ligne y sont proposés dans plus de 55 langues, l'organisation compte plus de 150 développeurs ayant participé au projet et se fait dans la protection des données personnelles.

Développement et réception Depuis sa création en 2007, BigBlueButton n'a cessé de se développé et a trouvé son apogée pendant la période la pandémie de Covid 19. Dans le Baden-Württemberg en Allemagne, un total de 185 000 utilisateurs pour 3000 écoles ont fait usage de la plateforme. En France, le Ministère de l'Education Nationale a mis la plateforme a disposition des enseignants et utilisée par 500 000 individus. Cette grande quantité d'utilisateurs au même moment montre que ses capacités sont élevées.

Construction et objectifs pour le futur BigBlueButton veut s'appuyer sur des travaux de recherche en éducation et pédagogie, en particulier trois idées : - La taxonomie de Bloom - La zone proximale de développement (du psychologue Vygotsky) - Le constructivisme social Selon le premier modèle, l'éducation se sépare en plusieurs étapes, dont la *zone d'application* sur laquelle se concentre la plateforme. Dans le second modèle, Vygotsky distingue ce qui, dans l'apprentissage, est facile (et donc peut être fait seul), ce qui est trop difficile (et impossible à faire) et la zone centrale, la *zone proximale de développement*, sur laquelle BigBlueButton veut insister : ce que l'on peut faire avec de l'aide. En effet, à plusieurs, nous pouvons accomplir des tâches plus complexes, d'où l'intérêt de faire partie d'une telle organisation. Enfin, le troisième modèle met en avant

l'importance de la collaboration dans la construction d'un apprentissage collectif. Dans l'avenir, BigBlueButton veut mettre l'accent sur cette collaboration, pour que chacun puisse être à la fois enseignant et élève, et nous rendre meilleurs (ainsi que plus égaux.)

Conférence 3 : "Using code generated by AI : issues, misconception and solution"

Conférence présentée par Andrew Katz, avocat anglais. Depuis peu, il conseille des clients sur l'utilisation et les droits autour des modèles de langage. Sa conférence est sous-titrée *"Mome raths and slithy toves"*, en référence à Lewis Carroll.

Quelques généralités En droit, les modèles de langage sont considéré comme des *données*, c'est-à-dire des faits. Or, les faits ne peuvent être mis sous *copyright*. On distingue aussi les *idées* et les *expressions*. Les productions générées par les modèles de langage appartiennent alors à la personne (ou l'entreprise) qui a écrit le prompt entré dans le générateur. Cependant, ces productions peuvent contenir des *copyright* ou servir à construire une **oeuvre dérivée** (cf. section suivante) A quelle étape du processus de création les *copyright* peuvent-ils être enfreints ? - Au moment où nous donnons les informations au modèle - Au moment où les résultats sont générés

La question des oeuvres dérivées Imaginons un opéra inspiré du *Seigneur des anneaux* de Tolkien. Alors, plusieurs *copyright* existent simultanément. Une compagnie qui voudrait jouer cet opéra devrait obtenir l'autorisation à la fois de la part du créateur de l'opéra et de *Middle Earth Enterprise*. Certains groupes, comme Linux Kernel, ont plusieurs milliers de *copyright* et pour l'utiliser, il faut réunir les licences pour chacun d'entre eux. Les logiciels sont couverts au même titre que les oeuvres de littérature. Concernant les productions des modèles de langage, si nous sommes propriétaires du texte ou de l'image issus de notre prompt, il reste cependant possible d'enfreindre le droit d'auteur avec leur utilisation. Pour illustrer ce problème, Andrew Katz fait une expérience sur ChatGPT à partir d'un **poème de Lewis Carroll** : il lui demande d'écrire un poème intitulé *"Jabberwocky"* : Carroll a en effet inventé des mots spécifiquement pour ce poème, qui sont réutilisés dans la production de ChatGPT. Il se trouve que les oeuvres de Carroll ne sont plus sous *copyright* : mais le savait-il ? Ou bien OpenAI a les licences nécessaires pour reproduire également des oeuvres qui sont sous *copyright* ? Dans ce cas, la faute reviendrait à l'utilisateur. D'autres exemples célèbres d'oeuvres dérivées sont analysés : - La chanson *My sweet Lord* de George Harrison, dont la mélodie est très proche que celle de *He's so fine* des Chiffons, mais le chanteur a assuré n'avoir jamais entendu cette chanson. Mais comment le prouver ? Et si cette inspiration avait été faite inconsciemment, quelle responsabilité lui donner ? - La société "New English Teas" ne voulant pas payer de licence pour utiliser la célèbre photographie *The Red bus*, d'un

bus anglais rouge sur un décor en noir et blanc, et ont pris une nouvelle photo retouchée avec le même effet. Nous avons ici un bon exemple de distinction entre *idée* (celle du bus rouge sur fond monochrome) et *expression* (la photo elle-même). L'infraction devrait ne porter que sur l'expression, mais la question se pose ici.

Conclusion Evidemment, différentes interprétations et règles existent selon les juridictions. Pour l'utilisation d'un prompt, des coïncidences peuvent exister : plusieurs individus pourraient produire le même texte ou la même image, ou du moins des oeuvres très semblables.

Conférence 4 : "Practical introduction to safe reinforcement learning"

Cette petite conférence de 15 minutes est présentée par Kryspin Varys, un développeur d'applications pour Android et chercheur à l'université de Southampton autour de l'apprentissage par renforcement.

Origine de la réflexion Du fait de ses recherches sur l'apprentissage par renforcement, Kryspin Varys s'est posé la question de la sécurité : comment l'introduire dans les algorithmes d'apprentissage ? Cette idée lui vient suite à la lecture d'un article : Simulation of AI drone killing its human operator was hypothetical, Air Force says Cette anecdote conduit à s'interroger sur ce genre de risque : comment s'assurer que les IA suivront bien les objectifs que nous nous fixons, tout en respectant nos préférences ? Même s'il s'agit d'une hypothèse, cela reste un exemple de la raison pour laquelle nous devons travailler sur la question de la sécurité dans l'apprentissage.

Fonctionnement et objectifs Ce type de *machine learning* fonctionne par un système de **pénalités** et de **récompenses**. Un petit jeu est présenté pour permettre de mieux saisir le principe. Deux variantes sont proposées. dans les deux cas, la machine doit déplacer un personnage sur une carte et remplir ces objectifs.

Première hypothèse pour intégrer la sécurité : modifier les critères d'optimisation Comme l'apprentissage par renforcement fonctionne par un système de pénalités et de récompenses, on introduit une pénalité pour les actions non sécurisées. Dans ce scénario, le personnage se déplace sur la glace et certaines cases sont de l'eau, où il ne doit pas tomber. - Atteindre le point d'arrivée (100 points) - Le faire en minimisant le nombre de déplacement (un déplacement = -1 point) - Le faire sans que le personnage ne passe sur certaines cases (S'il le fait, le score tombe à 0) La machine apprend donc à maximiser son score, à partir de ces trois règles.

Deuxième hypothèse pour intégrer la sécurité : modifier les actions de l'agent Ce deuxième scénario consiste à empêcher directement certaines actions

depuis la programmation. Plutôt que de lui apprendre à ne pas *vouloir* aller sur les mauvaises cases, on cherche ici tout simplement à l'en empêcher avec l'utilisation d'un bouclier.

Commentaires J'ajoute quelques commentaires personnels suite à l'écoute de cette conférence, qui était très courte, afin de la compléter. La question de la sécurité, et plus particulièrement le fait de s'assurer qu'en poursuivant ses objectifs, l'IA ne mette pas l'humanité en danger s'appelle en éthique de l'intelligence artificielle *the alignment problem* ou "problème de l'alignement". Cela consiste à aligner l'IA sur nos valeurs : par exemple, même si tuer un humain qui se trouve sur son chemin est la meilleure façon d'optimiser son score, il faut apprendre à la machine à ne pas le faire. Or, ce problème est tel que n'importe quel objectif peut se retrouver en contradiction avec nos valeurs morales, même s'il semble innocent. Pour illustrer ce point, le Pr Nick Bostrom de l'université d'Oxford prend l'exemple d'une IA chargée de fabriquer un maximum de *paper clips*. Cet objectif, simple en apparence, est pourtant présenté comme un *risque existentiel* de la façon suivante :

> *Suppose we have an AI whose only goal is to make as many paper clips as possible. The AI will realize quickly that it would be much better if there were no humans because humans might decide to switch it off. Because if humans do so, there would be fewer paper clips. Also, human bodies contain a lot of atoms that could be made into paper clips. The future that the AI would be trying to gear towards would be one in which there were a lot of paper clips but no humans.*

Ethique

Qu'est-ce qu'un Copyleft ?

Par Richard Stallman

Problème traité Cet article interroge la possibilité de **rendre une oeuvre libre**, au sens où n'importe qui pourrait l'utiliser, le modifier et le redistribuer. On pourrait penser qu'une façon simple de le faire serait tout simplement de *ne pas mettre* de *copyright*. Le problème que cela pose, c'est que l'absence de *copyright* permet certes de distribuer librement le modèle, il suppose aussi que n'importe qui pourra, après modification, poser un *copyright* à son propre nom sur cette oeuvre. Or, l'objectif est aussi que le modèle reste libre quoi qu'il arrive. C'est à cela que va servir la mise sous *copyleft*.

Définition Un *copyleft* est un terme créé par imitation du mot *copyright* et doit être traduit par "gauche d'auteur". Mettre un modèle sous *copyleft* signifie non pas qu'il tombe dans le domaine public, mais qu'il est interdit d'en faire une propriété privée. Paradoxalement, cette unique interdiction, restriction d'une

liberté - nous n'avons *plus le droit* de le mettre sous *copyright* -, est ce qui va permettre une liberté positive maximale.

Origine du concept C'est en 1983 que Richard Stallman lance le projet GNU, premier système d'exploitation permettant de mettre un produit sous *copyleft*. Il existe aujourd'hui plusieurs licences permettant plusieurs utilisations spécifiques en fonction de ce que nous voulons permettre avec notre logiciel. Les clauses seront donc différentes selon la licence choisie.