

LLMs & Pédagogie

Exemples (très) simples d'utilisation en 2023
pour l'enseignement de l'informatique

Emmanuel Viennet, Université Sorbonne Paris Nord

Je suis expert en IA (haha)

 ChatGPT



Voici deux images de style cartoon représentant un homme extrêmement âgé et sage, vu de profil et tourné vers la droite.

UNIVERSITE DE PARIS-SUD
CENTRE D'ORSAY

THESE

présentée
pour obtenir

LE TITRE DE DOCTEUR EN SCIENCES

Spécialité Informatique

par

Emmanuel VIENNET

Architectures Connexionnistes Multi-Modulaires
Application à l'Analyse de Scène

Soutenue le **18 juin 1993** devant la commission d'examen composée de

M.	Yves Kodratoff	President
M.	Christoph von der Malsburg	Rapporteur
M.	Bernard Victorri	Rapporteur
Mme.	Françoise Fogelman	Examineur
M.	Michel Weinfeld	Examineur

~~Je suis expert en IA~~

non, je travaille depuis 30 ans sur l'apprentissage automatique, les réseaux de neurones

mais comme vous j'ai été surpris par les performances de modèles de génératifs (couplés à l'apprentissage par renforcement)

Et ne les utilise pas mieux que vous 😊

UNIVERSITE DE PARIS-SUD
CENTRE D'ORSAY

THESE

présentée
pour obtenir

LE TITRE DE DOCTEUR EN SCIENCES

Spécialité Informatique

par

Emmanuel VIENNET

Architectures Connexionnistes Multi-Modulaires
Application à l'Analyse de Scène

Soutenu le 18 juin 1993 devant la commission d'examen composée de

M.	Yves Kodratoff	President
M.	Christoph von der Malsburg	Rapporteur
M.	Bernard Victorri	Rapporteur
Mme.	Françoise Fogelman	Examineur
M.	Michel Weinfeld	Examineur

1. Utilisation de base pour un enseignant

Objectifs:

- préparer plus rapidement un cours ou des exercices
- générer des quiz variés (pour entraîner les étudiants ou pour réduire la fraude lors des évaluations)

Exemple: un cours sur les graphes

EM

You

Donne moi le plan d'un cours d'une heure d'introduction à la théorie des graphes, pour des informaticiens préparant un bachelor.

Indique pour chaque section le temps indicatif à y consacrer.

Formate ta réponse en markdown.

Plan de Cours: Introduction à la Théorie des Graphes

I. Introduction (10 minutes)

- **Présentation du cours** (2 minutes)
 - Objectifs du cours
 - Importance de la théorie des graphes en informatique
- **Définition d'un graphe** (3 minutes)
 - Graphes dirigés et non dirigés
 - Exemples de graphes dans la vie réelle et en informatique
- **Terminologie de base** (5 minutes)
 - Sommets (nœuds)
 - Arêtes (liens)
 - Degré d'un sommet
 - Chemins et circuits

II. Représentation des Graphes (10 minutes)

- **Matrice d'adjacence** (5 minutes)
 - Définition et exemples
 - Avantages et inconvénients
- **Listes d'adjacence** (5 minutes)
 - Définition et exemples
 - Comparaison avec la matrice d'adjacence

III. Types de Graphes (10 minutes)

- **Graphes simples, multigraphes, et graphes pondérés** (5 minutes)
 - Définitions et caractéristiques
 - Exemples et applications
- **Graphes spéciaux** (5 minutes)
 - Graphes complets, bipartis, arborescences
 - Applications en informatique

IV. Parcours de Graphes (15 minutes)

- **Parcours en largeur (Breadth-First Search - BFS)** (7 minutes)
 - Algorithme et exemple
 - Applications et propriétés
- **Parcours en profondeur (Depth-First Search - DFS)** (8 minutes)
 - Algorithme et exemple
 - Applications et propriétés

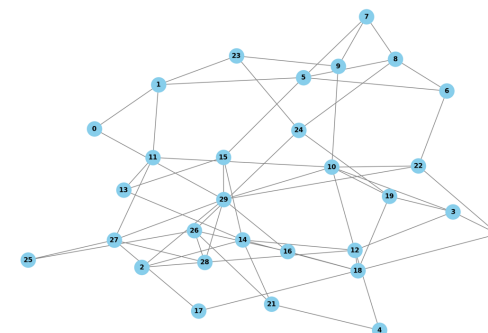
V. Applications de la Théorie des Graphes en Informatique (10 minutes)

- **Réseaux sociaux et analyse de réseaux** (3 minutes)
 - Exemples concrets
- **Algorithmes de cheminement** (3 minutes)
 - Plus court chemin, algorithmes de Dijkstra et Floyd-Warshall
- **Problèmes classiques** (4 minutes)
 - Coloration de graphes, problèmes de couplage, problèmes de flot maximum

VI. Conclusion et Ressources pour Approfondir (5 minutes)

- **Résumé du cours** (2 minutes)
- **Ressources supplémentaires** (2 minutes)
 - Livres, cours en ligne, articles
- **Questions et réponses** (1 minute)

Small-World Network Graph



Codage d'une classe de graphe

EM You

You will give an introduction course on graph theory. First, provide an implementation of graph in pure Python, using adjacency lists.

Methods to add a node, an edge, access to nodes and edges, neighbors of a node.

```
python Copy code

class Graph:
    def __init__(self):
        self.adjacency_list = {}

    def add_node(self, node):
        if node not in self.adjacency_list:
            self.adjacency_list[node] = []

    def add_edge(self, node1, node2):
        if node1 in self.adjacency_list and node2 in self.adjacency_list:
            self.adjacency_list[node1].append(node2)
            self.adjacency_list[node2].append(node1) # Assuming it's a

    def get_nodes(self):
        return list(self.adjacency_list.keys())

    def get_edges(self):
        edges = []
        for node, neighbors in self.adjacency_list.items():
            for neighbor in neighbors:
                if (neighbor, node) not in edges: # Avoid duplicates
                    edges.append((node, neighbor))
        return edges
```

```
def get_neighbors(self, node):
    return self.adjacency_list[node] if node in self.adjacency_list

def __str__(self):
    return str(self.adjacency_list)

# Example usage
g = Graph()
g.add_node("A")
g.add_node("B")
g.add_node("C")
g.add_edge("A", "B")
g.add_edge("B", "C")

print("Nodes:", g.get_nodes())
print("Edges:", g.get_edges())
print("Neighbors of B:", g.get_neighbors("B"))
```

Codage d'une classe de graphe

La solution proposée n'est pas mal, mais...



You

get_edges is checking for duplicates. I think it would be better to check once, in add_edge. What do you think ?



ChatGPT

You're absolutely right. Modifying the `add_edge` method to prevent duplicate edges is more efficient and ensures the graph maintains a consistent state. By handling duplicates during edge insertion, we can simplify the `get_edges` method since it no longer needs to check for duplicates.

Here's an updated implementation with the duplicate check in the `add_edge` method:

python

Copy code

```
class Graph:
    def __init__(self):
        self.adjacency_list = {}

    def add_node(self, node):
        if node not in self.adjacency_list:
            self.adjacency_list[node] = []

    def add_edge(self, node1, node2):
        if node1 in self.adjacency_list and node2 in self.adjacency_list:
            if node2 not in self.adjacency_list[node1]:
                self.adjacency_list[node1].append(node2)
            if node1 not in self.adjacency_list[node2]: # For undirected
                self.adjacency_list[node2].append(node1)
```

Codage d'une classe de graphe (solution 2)

python

Copy code

```
class Graph:
    def __init__(self):
        self.adjacency_list = {}

    def add_node(self, node):
        if node not in self.adjacency_list:
            self.adjacency_list[node] = []

    def add_edge(self, node1, node2):
        if node1 in self.adjacency_list and node2 in self.adjacency_list:
            if node2 not in self.adjacency_list[node1]:
                self.adjacency_list[node1].append(node2)
            if node1 not in self.adjacency_list[node2]: # For undirected
                self.adjacency_list[node2].append(node1)

    def get_nodes(self):
        return list(self.adjacency_list.keys())

    def get_edges(self):
        edges = []
        for node, neighbors in self.adjacency_list.items():
            for neighbor in neighbors:
                if (neighbor, node) not in edges: # Still needed to avoid
                    edges.append((node, neighbor))
        return edges
```

```
def get_neighbors(self, node):
    return self.adjacency_list[node] if node in self.adjacency_list

def __str__(self):
    return str(self.adjacency_list)

# Example usage
g = Graph()
g.add_node("A")
g.add_node("B")
g.add_node("C")
g.add_edge("A", "B")
g.add_edge("B", "C")
g.add_edge("A", "B") # Trying to add a duplicate edge

print("Nodes:", g.get_nodes())
print("Edges:", g.get_edges())
print("Neighbors of B:", g.get_neighbors("B"))
```

Codage d'une classe de graphe

Le dialogue d'un **expert** avec avec le LLMs permet d'arriver rapidement à un code propre et fonctionnel.

Mais le **débutant** en informatique n'aurait pas vu le problème (ici l'enjeu est d'apprendre à coder en ayant toujours présent à l'esprit la complexité du traitement)

D'ailleurs ChatGPT s'embrouille aussi facilement qu'un étudiant débutant:

You
For a graph with N nodes and L edges, what is the complexity of `get_edges` ?

ChatGPT
The complexity of the `get_edges` method in the provided graph implementation can be analyzed based on the number of nodes N and the number of edges L in the graph.

The `get_edges` method iterates over each node and its adjacent nodes (neighbors) to compile a list of edges. Let's break it down:

- Iterating Over Nodes:** The method iterates over each of the N nodes in the graph.
- Iterating Over Neighbors:** For each node, it iterates over its neighbors. The total number of neighbor iterations across all nodes is proportional to $2L$, because each edge in an undirected graph will be encountered twice – once for each of its two nodes.
- Checking for Duplicates:** The method checks if the edge (neighbor, node) is already in the list of edges to avoid duplicates. This check is performed for each edge and has a complexity of $O(1)$ per check because it's just a membership check in a list.

Putting it all together, the overall complexity of the `get_edges` method is $O(N + 2L)$. However, since L (the number of edges) typically dominates N (the number of nodes) in most practical scenarios (especially in dense graphs), the complexity is often approximated as $O(L)$.

You
Why do you say that

This check is performed for each edge and has a complexity of $O(1)$ per check because it's just a membership check in a list.

ChatGPT
I apologize for the confusion in my previous explanation. You are correct to question the statement about the complexity of the membership check in a list being $O(1)$. In fact, checking for membership in a list has a complexity of $O(L)$ in the worst case, where L is the length of the list. This is because, in the worst case, the check might need to compare the item with every element in the list until it finds a match or reaches the end of the list.

In the context of the `get_edges` method, this has a significant impact on the overall complexity:

- Iterating Over Nodes:** Iterating over each of the N nodes is $O(N)$.
- Iterating Over Neighbors:** For each node, iterating over its neighbors is proportional to $2L$ in total, as each edge is counted twice in an undirected graph.
- Checking for Duplicates:** The duplicate check, which is a membership check in the list of edges, is $O(L)$ for each check in the worst case. Since this check is inside the loop over the neighbors, it significantly increases the complexity.

Given this, the overall complexity of the `get_edges` method is more accurately described as $O(N \times L)$. This is because, for each of the N nodes, it potentially performs an $O(L)$ operation (checking if an edge is in the list of edges).

Conclusions 1

On le savait déjà:

- ChatGPT est excellent pour générer du texte ou du code
- il peut se tromper ou proposer des solutions imparfaites
- il peut se corriger en dialoguant avec un « expert »

2. Avec les étudiants

Objectifs:

- détecter les « mauvais » usages de ChatGPT
- critiquer les solutions avec les étudiants
- apprendre aux étudiants à *prompter* correctement

Emmanuel Viennet emmanuelviennet@gmail.com

Oct 17, 2023, 7:51 AM

to emmanuel.viennet

Cher Professeur Viennet,

J'espère que vous vous trouvez en bonne santé. Je me permets de vous écrire au sujet de ma note pour l'Examen final Service Réseaux Avancé que j'ai récemment reçue. Après avoir pris connaissance de ma note, j'ai des préoccupations que je souhaiterais discuter avec vous.

Tout d'abord, je tiens à exprimer ma gratitude pour le temps et les efforts que vous avez investis dans l'enseignement de ce cours. Votre expertise et votre dévouement ont grandement contribué à ma compréhension du sujet, et je suis reconnaissant pour l'opportunité de suivre votre cours.

Cependant, après avoir examiné attentivement ma note(12.22), j'ai remarqué certaines incohérences par rapport à mes attentes en termes de performance.

J'aimerais donc solliciter une révision de ma note pour ce cours. Je suis prêt à fournir des explications détaillées et des justifications concernant les éléments pour lesquels je crois que ma note peut être révisée à la hausse. Je comprends que vous

Bonjour,

j'ai bien reçu votre message et je trouve qu'il ressemble à un message écrit par une IA (ChatGPT ?), est-ce que je me trompe ?

Votre note n'est pas 12.22 comme vous l'indiquez:

R303 - Services réseaux avancés	^
Examen écrit	18.89 Coef. 04.00

Non monsieur vous ne vous trompez pas je l'avais bien écrit avec une IA. Je m'en excuse.
Effectivement vous avez raison ie me suis trompé de compte ie suis rentré dans le compte d'i

Que fait l'étudiant de 2023 ?

EXERCICE 4 - Envoi automatisé de messages

1- Écrire un script en langage Python qui envoie à une adresse donnée un mail indiquant l'heure et la charge système (commande `w` ou pseudo-fichier `/proc/loadavg`).



ChatGPT

Pour réaliser cet exercice, nous allons écrire un script Python qui effectue les opérations suivantes :

1. Récupérer l'heure actuelle et la charge système.
2. Envoyer un email à une adresse spécifiée contenant ces informations.

Nous utiliserons la bibliothèque `smtplib` pour envoyer l'email et `email` pour construire le message. Pour obtenir l'heure et la charge système, nous pouvons utiliser les modules `datetime` et `os` respectivement. Voici comment le script pourrait se présenter :

```
python Copy code

import smtplib
import os
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from datetime import datetime

def get_system_load():
    # Lecture du fichier /proc/loadavg pour obtenir la charge système
    with open('/proc/loadavg', 'r') as file:
        load_avg = file.read().strip()
    return load_avg
```

```
def send_email(receiver_address, subject, body):
    # Informations de connexion pour le serveur SMTP
    smtp_server = 'smtp.example.com' # Remplacer par le serveur SMTP
    smtp_port = 587 # Port pour le serveur SMTP
    smtp_user = 'your-email@example.com' # Remplacer par votre adresse
    smtp_password = 'your-password' # Remplacer par votre mot de passe

    # Construction du message
    message = MIMEMultipart()
    message['From'] = smtp_user
    message['To'] = receiver_address
    message['Subject'] = subject
    message.attach(MIMEText(body, 'plain'))

    # Connexion au serveur SMTP et envoi de l'email
    with smtplib.SMTP(smtp_server, smtp_port) as server:
        server.starttls()
        server.login(smtp_user, smtp_password)
        server.sendmail(smtp_user, receiver_address, message.as_string())

if __name__ == "__main__":
    receiver = 'destinataire@example.com' # Remplacer par l'adresse du
    current_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    system_load = get_system_load()
    subject = "Rapport d'heure et de charge système"
    body = f"Heure actuelle: {current_time}\nCharge système: {system_lo

    send_email(receiver, subject, body)
```

Que fait l'étudiant de 2023 ?

EXERCICE 4 - Envoi automatisé de messages

1- Écrire un script en langage Python qui envoie à une adresse donnée un mail indiquant l'heure et la charge système (commande `w` ou pseudo-fichier `/proc/loadavg`).

La solution (avec explications et commentaires) de ChatGPT est parfaite, mais:

- le cours n'a pas encore abordé la sécurisation (port 587 au lieu de 25)
- le format `MIMEMultipart` n'a pas non plus été abordé
- de plus, l'étudiant doit renseigner correctement les variables de configuration (que ChatGPT a pertinemment commenté)

Que fait l'étudiant de 2023 ?



You

Utilise le protocole SMTP standard (non sécurisé) et un message simple (sans multipart)



ChatGPT

D'accord, je vais simplifier le script pour utiliser le protocole SMTP standard (non sécurisé) et pour envoyer un message simple sans utiliser `MIMEMultipart`. Voici le script modifié :

python

Copy code

```
import smtplib
import os
from email.mime.text import MIMEText
from datetime import datetime

def get_system_load():
    # Lecture du fichier /proc/loadavg pour obtenir la charge système
    with open('/proc/loadavg', 'r') as file:
        load_avg = file.read().strip()
    return load_avg

def send_email(receiver_address, subject, body):
    # Informations de connexion pour le serveur SMTP
    smtp_server = 'smtp.example.com' # Remplacer par le serveur SMTP r
    smtp_port = 25 # Port standard pour SMTP non sécurisé
    smtp_user = 'your-email@example.com' # Remplacer par votre adresse
    smtp_password = 'your-password' # Remplacer par votre mot de passe

    # Création du message
    message = f"Subject: {subject}\n\n{body}"

    # Connexion au serveur SMTP et envoi de l'email
    with smtplib.SMTP(smtp_server, smtp_port) as server:
        server.login(smtp_user, smtp_password)
        server.sendmail(smtp_user, receiver_address, message)
```


Conclusions 2

- ChatGPT propose souvent des solutions trop « parfaites » aux exercices simples de codage de niveau Bachelor ou Master
- ... ce qui permet à l'enseignant de détecter aisément le copier/coller idiot de solutions.
- L'étudiant doit avoir compris son cours pour utiliser utilement ChatGPT
- Activité en TP: obtenir un code complet et propre en n'utilisant QUE ChatGPT.



3. L'étudiant seul

Objectifs:

- s'entraîner
- réviser
- réussir l'examen facilement si le prof l'a rédigé avec ChatGPT 😊

Digérer un cours



You

upload des diapos du cours en pdf



R303-03-Cours-Mail.pdf
PDF

Voici un cours sur la messagerie. Résumer les points clés en quelques courts paragraphes



ChatGPT

Le cours sur la messagerie couvre plusieurs aspects clés des services de messagerie électronique. Il aborde les protocoles SMTP (Simple Mail Transfer Protocol) pour l'envoi de courriels, ainsi que IMAP (Internet Message Access Protocol) et POP3 (Post Office Protocol 3) pour la réception des courriels. Le cours explique le rôle des agents utilisateur de messagerie (MUA), des agents de transfert de courrier (MTA) et des agents de distribution de courrier (MDA). Il souligne également l'importance de sécuriser les services de messagerie, notamment par l'utilisation de SMTPS (SMTP Secure) et IMAPS (IMAP Secure), qui chiffrent les communications pour garantir l'intégrité, la confidentialité et l'authentification des données. Le cours mentionne aussi le service de webmail, qui permet l'accès aux e-mails via un navigateur web.

Pour apprendre: digérer un cours, s'entraîner



You

Je dois apprendre ce cours. Pose moi 8 questions sur ce que je devrais savoir.



ChatGPT

Voici huit questions pour tester votre compréhension du cours sur la messagerie électronique :

1. Quel est le rôle principal du protocole SMTP dans le système de messagerie électronique ?
2. Comment les protocoles IMAP et POP3 diffèrent-ils dans la gestion des e-mails ?
3. Quelle est la différence entre un agent utilisateur de messagerie (MUA) et un agent de transfert de courrier (MTA) ?
4. Comment SMTPS améliore-t-il la sécurité par rapport au SMTP standard ?
5. En quoi consiste un agent de distribution de courrier (MDA) et quel est son rôle ?
6. Quels sont les avantages de l'utilisation du service de webmail par rapport aux clients de messagerie traditionnels ?
7. Comment IMAPS assure-t-il une meilleure sécurité des communications par rapport à IMAP standard ?
8. Quelles mesures peuvent être prises pour garantir l'intégrité, la confidentialité et l'authentification des données dans les services de messagerie électronique ?

Ces questions couvrent les principaux thèmes du cours et vous aideront à mieux comprendre les différents aspects des services de messagerie électronique.



Conclusions provisoires

- ChatGPT est évidemment très utile aux enseignants comme aux étudiants
- Quoi enseigner ?

Moins de syntaxe (il le fait très bien), plus de « sémantique »

- Les « experts » bénéficient plus de l'IA que les débutants (?)
- Entraîner nos étudiants à l'utiliser et le critiquer

*textes et illustrations par
ChatGPT 3.5 ou 4*

 **You**
un cartoon représentant ChatGPT en train d'enseigner à un groupe d'étudiants

 **ChatGPT**

Hmm...something seems to have gone wrong.

EM

You

Explique comment utiliser ChatGPT en classe à l'université avec des étudiants d'informatique



ChatGPT

Utiliser ChatGPT dans un cadre universitaire, notamment dans des cours d'informatique, peut être très bénéfique pour enrichir l'expérience d'apprentissage des étudiants. Voici plusieurs façons d'intégrer ChatGPT en classe :

1. **Explication de Concepts Complexes** : ChatGPT peut être utilisé pour expliquer des concepts informatiques complexes de manière interactive. Les étudiants peuvent poser des questions spécifiques et obtenir des réponses claires et détaillées.
2. **Assistance aux Devoirs** : ChatGPT peut aider les étudiants à comprendre les devoirs ou les projets. Il peut fournir des conseils sur la résolution de problèmes, la programmation, et d'autres aspects techniques.
3. **Débats et Discussions de Groupe** : Intégrez ChatGPT dans les discussions de groupe pour apporter un éclairage supplémentaire ou pour poser des questions qui stimulent la réflexion et le débat.
4. **Exercices de Codage** : Utilisez ChatGPT pour aider les étudiants dans leurs exercices de programmation. Il peut offrir des suggestions sur la syntaxe, la logique de programmation, et les meilleures pratiques.
5. **Études de Cas et Exemples** : ChatGPT peut générer des études de cas ou des exemples réels pour illustrer l'application des concepts théoriques.
6. **Apprentissage de Langages de Programmation** : Les étudiants peuvent interroger ChatGPT sur des langages de programmation spécifiques, leurs utilisations, et obtenir des exemples de code.
7. **Révision et Préparation aux Examens** : ChatGPT peut être un outil de révision efficace, fournissant des réponses aux questions de révision et aidant à clarifier les points de confusion.
8. **Feedback sur les Projets** : Les étudiants peuvent présenter leurs idées de projet à ChatGPT pour obtenir un retour initial, des suggestions d'amélioration, ou des perspectives alternatives.
9. **Sensibilisation à l'IA et à l'Éthique** : Discutez de l'utilisation de ChatGPT et d'autres IA, en abordant des sujets tels que l'éthique de l'IA, les biais des données, et la responsabilité dans le développement de l'IA.

Pour une utilisation optimale, il est important que les étudiants comprennent les limites de ChatGPT, notamment en termes de précision des informations et de mise à jour des connaissances. Encourager un esprit critique dans l'utilisation de cet outil est essentiel. De plus, intégrer ChatGPT dans le curriculum nécessite une planification et une adaptation par les enseignants pour s'assurer que l'outil complète efficacement les méthodes d'enseignement traditionnelles.

