Abstract

In this paper, we propose an anomaly detection methodology that aims to detect anomalies among complex data, such as images, and that is based on a confidence level rather than on a certain threshold or metric. In order to do that, we demonstrate the usefulness of using variational autoencoders (VAE) to deal with complex data and also to have some kind of representation from which we can apply hypothesis testings. From our experiments, we can show that our approach is able to detect images that are outliers in a given dataset by only specifying a certain confidence level. By using this approach, the anomaly detection becomes feasible for real-world complex images and is also easier to maintain and interpret considering the detection is made on a confidence level rather than on a vague and possibly changing metric.

Table des matières

Liste des tableaux

Liste des figures

<Dédicace si désiré>

Remerciements

<Texte des remerciements en prose.>

Introduction

La détection d'anomalie est un sujet complexe qui a généré beaucoup de littératures en statistiques, en apprentissage machine et plus récemment en vision numérique. Il existe plusieurs applications à ce sujet dans les domaines de la cyber-intrusion, de la finance et de l'assurance, de la médecine ou dans l'identification de dommages industriels (Chandola et al. (2007)). Une anomalie est définie par Zimek and Schubert (2017) comme un événement, une mesure ou une observation qui diffère significativement de la majorité des données. Une anomalie, ou également appelée une aberration, est un concept intrinsèque à plusieurs domaines reliés à l'analyse de données, car une telle donnée est généralement intéressante à identifier ou retirer d'une source de données. Il peut être intéressant de l'identifier simplement parce que c'est la tâche poursuivie. Il peut également être intéressant de la retirer avant de réaliser une autre tâche d'apprentissage.

Une difficulté reliée à la détection d'anomalie est qu'on doit souvent utiliser des données non étiquetées, car les anomalies sont généralement des données événements imprévues. Cela fait en sorte que le problème devient non-supervisé. Une autre difficulté, plus particulièrement reliée avec les approches non-supervisées, est qu'il faut généralement déterminer un seuil. Ce seuil nous permet de prendre une décision quant à la nature d'une donnée (anomalie ou non). Finalement, ces difficultés deviennent encore plus ardues lorsqu'on doit traiter avec des données complexes, comme des images.

Dans ce contexte de données à hautes dimensions, les réseaux de neurones sont couramment utilisés. En effet, leurs couches superposées peuvent partir d'une entrée complexe, comme une image, et compresser cette information vers une représentation plus simple et riche en informations. Les autoencodeurs sont une catégorie de réseaux de neurones fréquemment utilisés pour traiter un problème non-supervisée. Il existe d'ailleurs plusieurs applications d'autoencodeurs utilisées dans un contexte de détection d'anomalie où l'erreur de reconstruction est souvent utilisée comme indicateur d'anomalie. Cependant, ces méthodes requièrent de trouver un seuil, souvent sous forme de distance ou de métrique, qui peut être difficile à établir ou à expliquer. C'est d'ailleurs pour cette raison que An and Cho (2015) propose de se baser

sur une probabilité de reconstruction, qui est un mesure plus objective et ne requiert pas de seuil quelconque. Par contre, cette mesure de probabilité est basée sur la capacité de reconstruction, qui peut être problématique dans le contexte d'images complexes, plutôt que sur la représentation latente, qui elle est plus simple et compressée.

Dans cette étude, nous proposons une approche qui vise à simplifier la détermination du seuil nécessaire par rapport à la détection d'anomalie à partir de données non étiquetées. Avec cette contribution, nous proposons d'utiliser des méthodes existantes comme les autoencodeurs et les tests d'hypothèses pour encoder des structures de données complexes dans un cadre décisionnel simple et intuitif. À partir d'autoencodeurs variationnels (VAE), nous sommes en mesure de créer ces dites représentations et de détecter les anomalies à partir d'un niveau de confiance, plutôt que d'une métrique ou une distance quelconque.

Chapitre 1

Contexte

En premier lieu, nous allons faire un résumé de la théorie derrière les autoencodeurs et comment ceux-ci sont-ils pertinents dans un contexte de détection d'anomalies. Ensuite, nous allons décrire plus en détails un type particulier d'autoencodeur utilisé dans cette étude, soit l'autoencodeur variationnel. Finalement, nous allons couvrir quelques notions de base quant aux tests d'hypothèses, un cadre statistique classique pour prendre des décisions.

1.1 Les autoencodeurs

Un autoencodeur est un réseau de neurones qui a comme objectif d'apprendre une représentation intermédiaire et efficiente d'une entrée de manière non-supervisée (Goodfellow et al. (2016)). Pour réaliser cette objectif, l'autoencodeur se décompose en 2 composantes : un encodeur et un décodeur. L'encodeur reçoit en entrée x et convertit celui-ci vers une représentation latente z. Le décodeur prend en entrée cette représentation latente z et la décode pour ainsi retrouver le plus possible l'entrée initiale x. Cette structure de base est illustrée dans la figure 1.1. Historiquement, les autoencodeurs étaient vues comme une méthode de réduction de dimensionnalité, mais désormais ceux-ci ont davantage d'applications dû au fait qu'il peuvent apprendre des variables latentes riches en informations.

L'intuition derrière les autoencodeurs est essentiellement de reconstruire une entrée x en passant par 2 composantes ou fonctions (l'encodeur et le décodeur) apprises par le modèle. Ainsi, ce genre de méthode n'a pas besoin d'une étiquette y, car son objectif est basée sur x directement. C'est d'ailleurs pour cela qu'on parle d'une approche d'apprentissage non-supervisée. L'apprentissage des paramètres est fait en grande partie en minimisant l'erreur de reconstruction. La perte peut donc être définie par une fonction de la forme :

$$L(x, p_\phi\{q_\theta(x)\})$$

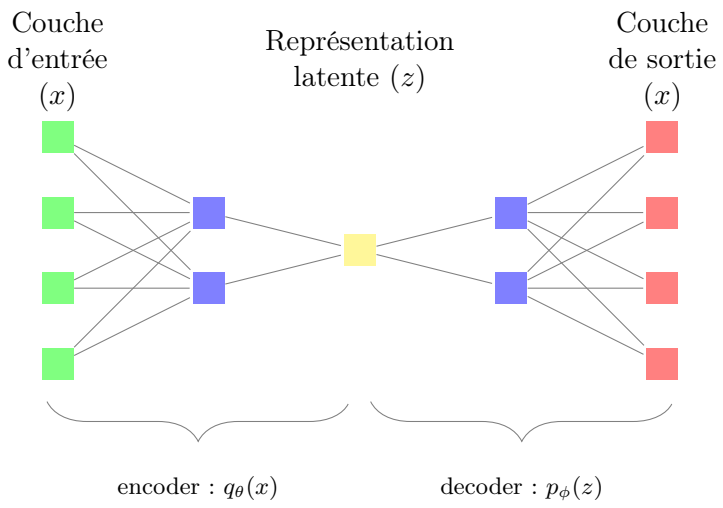

FIGURE 1.1 – Exemple illustrant la structure de base d'un autoencodeur. Dans le cas ci-dessus, on pourrait interpréter le schéma comme un réseau pleinement connecté où les blocs pourraient représentés les neurones et les liens seraient les poids, ou les paramètres du réseau. Le concept s'applique également à une architecture de réseau à convolutions, où les paramètres appris sont les filtres de convolutions.

où $q(x, \theta)$ est l'encodeur et $p(z, \phi)$ est le décodeur. L'optimisation de cette fonction de perte est faite par descente du gradient. En d'autres mots, les paramètres de l'encodeur et du décodeur sont fait graduellement en prenant la dérivée de la fonction de perte par rapport aux différents paramètres de ces deux composantes :

$$\Theta \leftarrow \Theta - \epsilon * \frac{\partial L}{\partial \Theta} \tag{1.1}$$

où ϵ est un taux d'apprentissage qui permet de moduler la vitesse d'apprentissage et $\Theta : \{\theta, \phi\}$ comprend les paramètres de l'encodeur et du décodeur.

Une fois que l'autoencodeur est entraîné adéquatement, il est possible d'utiliser l'erreur de reconstruction comme score d'anomalie. En effet, une donné mal reconstruite pourrait être vue comme une donnée aberrante, ou une donnée que le réseau n'a pas eu l'habitude de voir lors de l'entraînement. Dans Aggarwal (2016), ce genre de méthode de détection d'anomalies fait partie de la catégorie des algorithmes basés sur les modèles linéaires ou non-linéaires. Dans ce groupe de méthodes, on commence par ajuster un modèle linéaire ou non-linéaire et on utilise l'erreur de reconstruction, ou le résidu, comme indicateur d'anomalie. Les méthodes basées sur les régressions linéaires, l'analyse en composantes principales (PCA) or la factorisation de matrices font également parties de cette large catégorie de méthodes de détection d'anomalies. Par contre, la reconstruction n'est pas le seul critère qui peut être utilisé pour entraîner un autoencodeur. Dans la prochaine section, nous allons couvrir un type précis d'autoencodeur, soit l'autoencodeur variationnel. Nous verrons également quelle autre composante peut être

utilisée comme score d'anomalie.

1.2 Les autoencodeurs variationnels

Les autoencodeurs variationnels (VAE) Kingma and Welling (2013) ont une approche légèrement différente des autres autoencodeurs. En effet, au lieu d'encoder les données dans un vecteur de variables latentes à p dimensions, les données sont plutôt encodées dans 2 vecteurs de taille p : un vecteur de moyennes $\boldsymbol{\mu}$ et un vecteur de déviations standards $\boldsymbol{\sigma}$. Ces deux vecteurs sont ensuite utilisés pour générer la représentation latente à p dimensions, qui est en fait une simulation d'une loi normale multivariée avec les paramètres $\boldsymbol{\mu}$ et $\boldsymbol{\sigma}$. Cela permet donc d'obtenir pour une donnée x, une représentation latente continue. C'est d'ailleurs la particularité la plus importante des VAE par rapport aux autres autencodeurs. Dans d'autres mots, les autoencodeurs de base ont comme objectif d'apprendre une représentation latente qui pointe à quelque part dans cet espace latent, alors que les VAE apprennent une représentation qui pointe vers une zone de cette espace latent. Cette zone est effectivement définit par les paramètres $\boldsymbol{\mu}$ et $\boldsymbol{\sigma}$ associés à une entrée donnée. Cela veut aussi dire qu'une fois l'algorithme entraîné, la sortie de celui-ci est stochastique dans le sens où une entrée x peut donner 2 sorties différentes. Cependant, une donnée x va donner toujours les mêmes valeurs de $\boldsymbol{\mu}$ et $\boldsymbol{\sigma}$, cette composante n'est pas stochastique. La figure 1.2 illustre la structure de base des autoencodeurs variationnels. Dans la figure, on peut remarquer que les données en entrée commencent par passer par des couches cachées, qui peuvent être pleinement connectées ou de convolutions. Cette partie est l'encodeur ($q_\theta(x)$). Un peu avant d'arriver à la représentation latente, le réseau se divise en 2 composantes ($\boldsymbol{\mu}$ et $\boldsymbol{\sigma}$). La représentation latente z est ensuite générée en simulant d'une loi normale multivariée avec les paramètres correspondant aux couches précédentes. Une fois la représentation z générée, celle-ci est décodée jusqu'au format original par des couches pleinement connectées ou des couches de déconvolution. Cette partie est le décodeur ($p_\phi(z)$).

Les autoencodeurs variationnels sont également différents quant au calcul de perte qui est essentiel à l'optimisation du réseau. En effet, on ajoute une autre composante de perte à l'erreur de reconstruction de base. La fonction de perte est désormais définit comme une somme de deux composantes :

$$L(x, p_\phi\{q_\theta(x)\}) + D_{KL}\big[q_\theta(z|x)||p(z)\big]$$

où D_{KL} est une divergence de Kullback-Leibler. L'objectif de cette nouvelle composante est de s'assurer que la distribution encodée $q(z)$ et la distribution à priori $p(z)$ sont similaires. La distribution à priori $p(z)$ est une loi normale multivariée $N(0, I)$. Si cette composante de perte est suffisamment prise en compte lors de l'optimisation, nous devrions donc obtenir des paramètres μ et σ se rapprochant d'un vecteur de 0 et un vecteur de 1. Dans ce cas-ci,

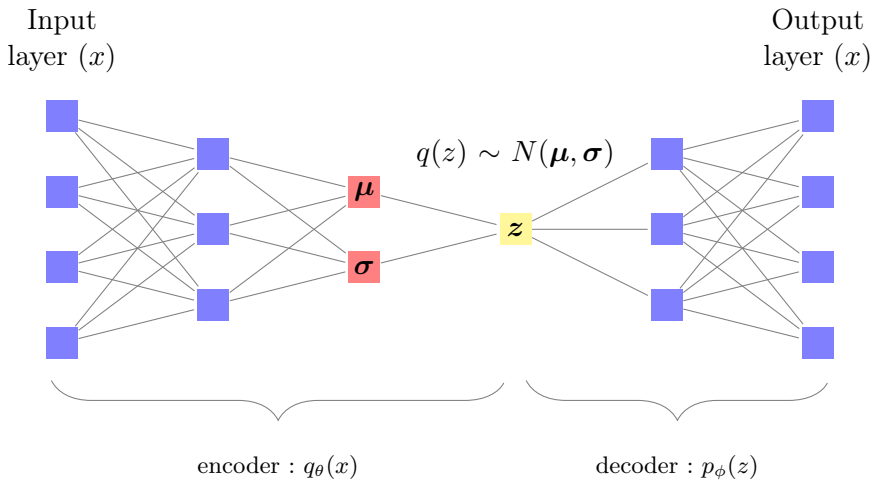

FIGURE 1.2 – Structure de base d'un autoencodeur variationnel. La représentation latente z est générée en simulant de la loi $q(z)$, qui suit une loi normale multivariée avec les paramètres μ et σ calculés par le réseau. C'est de là que vient la composante stochastique du modèle.

nous assumons l'indépendance entre les variables latentes, ce qui nous permet de définir σ comme un vecteur et non une matrice. Ce vecteur devient donc la diagonale de la matrice de variance-covariance.

Comme expliquée un peu plus tôt, la représentation latente de ce type d'autoencodeur est simulé d'une loi normale multivariée, donc stochastique. Cela pourrait en théorie rendre complexe la tâche d'optimisation du réseau. En effet, les paramètres du réseau, soit ceux de l'encodeur et du décodeur, sont optimisés par descente du gradient. On calcule donc la perte la perte $L(x)$ et on dérive cette fonction par rapport à chaque paramètres du réseau. Mais qu'en est-il de la loi normale multivariée qui à généré notre représentation latente ? Afin de simplifier le calcul des dérivées lors de la rétropropagation, on redéfinit le calcul de la représentation z dans un format plus simple. Cette simplification s'appelle la "reparametrization trick". En effet, il est possible pour certaines distributions, comme la loi normale, de séparer les paramètres (μ et σ) de la composante stochastique. Concrètement, on peut définir une simulation normale multivariée comme :

$$z = \mu + \sigma \odot \epsilon$$

où $\epsilon \sim N(0, 1)$. En bref, cela signifie que la couche z, illustrée dans la figure 1.2, est générée à partir de deux couches de paramètres μ et σ et d'une simulation $N(0, 1)$. Cela veut donc dire que la rétropropagation peut ignorer la composante stochastique et calculer les dérivées des couches de paramètres seulement, ce qui simplifie beaucoup le processus d'optimisation.

Au final, la composante de perte associée au critère de divergence de Kullbach-Leibler apporte

de la régularisation au modèle, en restreignant celui-ci dans un certain espace (Kingma and Welling (2013)). UN PEU PLUS DE JUICE SUR LA RÉGULARISATION.

1.3 Test d'hypothèses

Les tests d'hypothèse sont une méthode d'inférence statistique. L'objectif est de tester une hypothèse nulle (H_0) par rapport à une hypothèse alternative (H_1). Dans le contexte de détection d'anomalie, nous pourrions utiliser ce cadre d'inférence pour tester si une certaine observation provient d'une population attendue :

$\boldsymbol{H_0}$: x_i provient de la population P
$\boldsymbol{H_1}$: x_i ne provient pas de la population P

Une fois que nous avons définit une structure de test, soit l'hypothèse à confirmer ou infirmer, il faut se définir un cadre précis nous permettant de mettre en application ce test. Cela passe généralement par une hypothèse à priori à propos de notre échantillon de test. Cette hypothèse se décompose généralement en deux parties : une distribution attendue et une statistique de test. Avec ces deux composantes, on peut généralement calculer une valeur-p. La valeur-p est définit comme la probabilité, sous l'hypothèse nulle, d'observer la statistique de test de la distribution attendue. Quand la valeur-p est petite, cela signifie qu'il est peu probable d'observer l'observation sous l'hypothèse nulle. L'hypothèse nulle est généralement rejetée lorsque la valeur-p est plus petite qu'un certain seuil α. Ce seuil est également appelé le seuil de confiance. Lorsque l'hypothèse nulle est vraie et que la variable testée est continue, alors la distribution de cette valeur-p est distribuée de manière uniforme sur l'intervalle [0, 1]. Dans ce projet, un de nos objectif est de bénéficier de ce cadre d'inférence statistique en utilisant un seuil de confiance, plutôt qu'une métrique quelconque pour détecter les anomalies.

Chapitre 2

Méthodologie

Le coeur de notre travail consiste à proposer une méthode de détection d'anomalies où nous utilisons la représentation latente d'un autoencodeur variationnel qui sera transformée vers une métrique que nous pourrons tester via un test d'hypothèse. Ce test d'hypothèse nous permet donc de détecter des anomalies avec un niveau de confiance.

2.1 Les hypothèses de l'approche

Dans notre approche, nous supposons que nous avons accès à un jeu de données qui contient presque qu'entièrement des observations normales. En d'autres mots, il n'y pratiquement pas d'anomalies. Par contre, nous anticipons des entrées anormales dans le futur et nous voulons les détecter. Voici 2 exemples de situation où ce genre de contexte pourrait être plausible :

1. Une compagnie d'assurance propose maintenant à leurs assurés de prendre eux-mêmes des photos de leur véhicule accidenté et d'envoyer celles-ci à l'assureur. La compagnie s'attend à ce que les assurés puissent commettre des erreurs de manipulation en prenant les photos ou bien puissent envoyer les mauvaises photos. De son côté, l'assureur possède des photos de véhicules accidentés, prises par son personnel d'estimateurs, mais aucune photo erronée. Le jeu de données d'entraînement ne possède théoriquement aucune anomalie.
2. Une entreprise manufacturière qui produit des pièces de voiture vient de se procurer un nouveau système qui automatise davantage la production. L'entreprise veut faire un suivi des pièces produites par ce nouveau système via une caméra qui inspecte les produits finaux. L'entreprise possède déjà plusieurs images de pièces qui étaient produites avant l'acquisition du nouveau système. Le résultat final produit par ce nouveau système doit être identique à l'ancien. Par contre, il est difficile de prévoir de quoi auront l'air des défectuosités produites par ce nouveau système. Le jeu de données d'entraînement est donc constitué de plusieurs exemples normales, mais aucune anomalie.

2.2 Description de l'approche

Considérant que nous avons accès à un jeu de données ayant les caractéristiques décrites à la section précédente, nous proposons d'entraîner un autoencoder variationnel pour apprendre les caractéristiques, ou la distribution, de cette population "normale". Cette distribution apprise sera essentiellement contenu dans la représentation latente du VAE entraîné, ou plus spécifiquement dans les couches μ et σ du réseau. Comme nous l'avons vu dans la section 1.2, les VAE ont la particularité d'avoir une composante de perte associée à cette représentation latente, s'assurant ainsi que celle-ci s'approche d'une distribution à priori, soit une $N(0, I)$ dans notre cas. Une fois que l'autoencodeur est entraîné, nous pouvons utiliser la partie encodeur du réseau pour transformer chacune des instances de notre jeu de données d'entraînement vers des vecteurs μ et σ. Ces représentations latentes, encodées sous forme de vecteurs, contiennent l'information qui devrait permettre de savoir si une nouvelle instance partage ou non cette même information. Pour parvenir à faire cette conclusion sur une nouvelle instance, nous devons comparer l'information contenue de la représentation latente de cette instances avec toutes les représentations latentes de l'ensemble d'entraînement. Il est donc plus pratique d'agréger notre population d'entraînement en une seule représentation globale. Une première manière d'y arriver est de calculer un vecteur $\bar{\mu}$ et un vecteur $\bar{\sigma}$, qui sont simplement une moyenne sur la population d'entraînement. Supposons que notre jeu de données d'entraînement contient n observations et que nous encodons celles-ci vers des représentations latentes à k dimensions, l'élément i des vecteurs $\bar{\mu}$ et $\bar{\sigma}$ sera donné par :

$$\bar{\mu}^{(i)} = \frac{1}{n} \sum_{k=1}^{n} \mu^{(k)}$$
$$\bar{\sigma}^{(i)} = \frac{1}{n} \sum_{k=1}^{n} \sigma^{(k)}$$

Étant donné que les vecteurs $\bar{\mu}$ et $\bar{\sigma}$ sont essentiellement composé d'instances normales, on peut s'attendre que les vecteurs μ et σ de nouvelles instances normales soient relativement près de ces vecteurs moyen. À l'inverse, on peut s'attendre que ces 2 mêmes vecteurs, pour des instances anormales, ou des anomalies, soient plutôt loin de ces vecteurs moyens. Maintenant, comment savoir si la représentation latente d'une nouvelle instance est proche ou éloignée de cette représentation agrégée ? Étant donné que les vecteurs μ et σ représentent les paramètres d'une loi normale multivariée, nous pouvons utiliser la distance de Kullbach-Leibler pour quantifier le degré de similitude entre la distribution moyenne et la distribution μ et σ de la nouvelle instance. Cette même distance est d'ailleurs utilisée lors de l'optimisation pour calculer la composante de perte associée à la représentation latente. Finalement, il est possible d'utiliser les distances de KL calculées sur l'ensemble d'entraînement pour savoir si une nouvelle distance est élevée ou non. Cela devient possible étant donné que notre ensemble

d'entraînement contient presque uniquement des observations normales, donc des distances de KL qui devraient être petites. On peut d'ailleurs utiliser ces distances ordonnées pour avoir une mesure de probabilité d'une nouvelle instance. La méthodologie proposée pour calculer cette mesure de probabilité, ou cette valeur-p est résumé dans l'algorithme 1.

Algorithme 1 : Algorithme de détection d'anomalies basé sur un autoencodeur variationnel

Input : Ensemble de données d'entraînement sans anomalie $x^{(j)}, j = 1, ..., m$,
Ensemble de données de test avec anomalies $y^{(i)}, i = 1, ..., n$
Output : Valeurs-p pour chaque instance de test $p^{(i)}, i = 1, ..., n$
θ, $\phi \leftarrow$ paramètres de l'encodeur ($q_\theta(x)$) et du décodeur ($p_\phi(z)$) du VAE entraîné;
for *j=1 to m* **do**
  $\mu^{(j)} = p_\theta(x^{(j)})["mu"]$;
  $\sigma^{(j)} = p_\theta(x^{(j)})["sd"]$;
end
$\bar{\mu} = \frac{1}{m}\sum_{k=1}^{m} \mu^{(k)}$;
$\bar{\sigma} = \frac{1}{m}\sum_{k=1}^{m} \sigma^{(k)}$;
for *j=1 to m* **do**
  $kl\_train^{(j)} = kl\_distance(\mu^{(j)}, \sigma^{(j)}, \bar{\mu}, \bar{\sigma})$
end
$kl\_sorted = sort(kl\_train)$;
for *i=1 to n* **do**
  $\mu^{(i)} = p_\theta(y^{(i)})["mu"]$;
  $\sigma^{(i)} = p_\theta(y^{(i)})["sd"]$;
  $kl\_test^{(j)} = kl\_distance(\mu^{(i)}, \sigma^{(i)}, \bar{\mu}, \bar{\sigma})$;
  $p^{(i)} = rank_{kl\_sorted}(kl\_test)$ / m
end
return ***p***

Dans l'approche que nous proposons, la valeur-p est calculée à partir d'une distribution empirique, soit les distances de Kullbach-Leibler de toutes les instances du jeu de données d'entraînement. Au final, nous testons empiriquement si une nouvelle observation semble provenir de la population d'entraînement :

$$\boldsymbol{H_0} : y^{(i)} \text{ provient de la population } X$$
$$\boldsymbol{H_1} : y^{(i)} \text{ ne provient pas de la population } X$$

Puisque nous supposons que X, soit l'ensemble de données d'entraînement, est composé en grande partie d'observations normales, nous pouvons réécrire notre test comme étant :

$$\boldsymbol{H_0} : y^{(i)} \text{ n'est pas une anomalie}$$
$$\boldsymbol{H_1} : y^{(i)} \text{ est une anomalie}$$

Finalement, une fois que nous avons notre test statistique de définit, nous pouvons utiliser les valeurs-p calculées sur l'ensemble de test et conclure avec un niveau de confiance α si une instance est une anomalie (voir algorithme 2).

Algorithme 2 : Algorithme de prise de décision

Input : Valeurs-p pour les instances de test $p^{(i)}, i = 1, ..., n$,
niveau de confiance α
Output : indicateurs d'anomalies $o^{(i)}, i = 1, ..., n$
for *i=1 to n* **do**
  if $p^{(i)} < \alpha$ **then**
    $o^{(i)} = vrai$
  else
    $o^{(i)} = faux$
  end
end
return $\boldsymbol{o}$

2.3 Approche pour des données complexes

Parler de pourquoi utiliser les representation encoder pour le test au lieu de l'erreur de reconstruction. Peut-etre parler de la perceptual loss pour dealer avec une reconstruction difficile pour des images complex.

2.3.1 Représentation latentes

2.3.2 Perceptual loss

2.4 Avantages du test d'hypothèse

Parler du fait que c'est simple de definir un level of significance versus trouver une metric quelconque.

Chapitre 3

Expérimentations

3.0.1 Datasets

3.0.2 Models tested

3.0.3 Discussion

Conclusion

<Texte de la conclusion. Une thèse ou un mémoire devrait normalement se terminer par une conclusion placée avant les annexes, le cas échéant. La conclusion est traitée comme un chapitre normal, sauf qu'elle n'est pas numérotée.>

Annexe A

<Titre de l'annexe>

<Texte de l'annexe.>

Bibliographie

Aggarwal, C. C. (2016). *Outlier Analysis*. Springer Publishing Company, Incorporated, 2nd edition.

An, J. and Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability.

Chandola, V., Banerjee, A., and Kumar, V. (2007). Anomaly detection : A survey.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. cite arxiv :1312.6114.

Zimek, A. and Schubert, E. (2017). *Outlier Detection*, pages 1–5. Springer New York, New York, NY.