Classification de graphes par algorithmes génétiques et signatures de graphes. Application à la reconnaissance de symboles

Eugen Barbu, Romain Raveaux, Hervé Locteau, Sébastien Adam, Pierre Héroux, Eric Trupin

Classification de graphes par algorithmes génétiques et signatures de graphes. Application à la reconnaissance de symboles

E. Barbu[1] – R. Raveaux[1]–H. Locteau[1] – S. Adam[1] – P. Héroux[1] – E. Trupin[1]

[1] Laboratoire LITIS-PSI – Université de Rouen – UFR des Sciences et Techniques
Avenue de l'Université – 76800 Saint-Etienne du Rouvray – France

{Prénom.Nom}@univ-rouen.fr

Résumé *: Nous présentons dans cet article une approche de classification de graphes. Le système proposé repose sur une mesure de dissimilarité originale calculée à partir de l'extraction de signatures de graphes (graph probing) et sur l'utilisation d'un algorithme génétique pour l'apprentissage de prototypes de graphes. Les prototypes appris peuvent différer des graphes de la base d'apprentissage d'origine et sont générés par l'application d'un algorithme d'optimisation génétique qui cherche à maximiser le taux de bonne reconnaissance. L'approche, générique, est ici appliquée à un problème de reconnaissance de symboles. Les tests sont réalisés en utilisant le benchmark proposé dans le cadre du concours de reconnaissance de symboles de la conférence GREC. Les résultats obtenus montrent l'intérêt de l'approche.*

Mots-clés *:* classification de graphes, mesures de dissimilarité entre graphes, signatures de graphes, algorithmes génétiques, reconnaissance de symboles

1 Introduction

Les graphes constituent un mode de représentation très fréquemment utilisé dans le domaine des sciences et technologies de l'information et de la communication. Ils permettent en effet de décrire naturellement dans un formalisme unifié des objets et les relations entre ces objets. C'est particulièrement le cas en reconnaissances de formes, et plus encore en reconnaissance de symboles. En effet, par nature, un symbole peut être décrit à partir des primitives le constituant (composantes connexes, occlusions, segments, arcs…) et des relations entre ces primitives (voisinage, connexions, parallélisme). On parle de méthodes structurelles de reconnaissance de symboles [LLA 01].

Dans un tel contexte de représentation structurelle, la reconnaissance de symboles segmentés est alors un problème de classification de graphes [SCH 03][SER 05]. Son objectif est d'affecter une classe à un graphe décrivant un symbole inconnu. Une base d'apprentissage contenant des exemples étiquetés est pour ce faire utilisée. Dans le contexte de la reconnaissance de symboles, cette base d'apprentissage se limite souvent à un unique exemple par classe puisqu'on ne connaît en général que le modèle idéal (souvent vectoriel) du symbole issu du logiciel de CAO [VAL 04].

Dans cet article, nous décrivons un système capable de classifier des graphes représentant des symboles en exploitant une base d'apprentissage composée d'un exemple par classe. Pour palier ce manque d'exemple, l'un des objectifs principaux que nous nous sommes fixé vise à ce que le système soit en mesure de prendre en compte la variabilité pouvant survenir dans la représentation structurelle générée à partir de l'image du symbole. Pour ce faire, à partir des N graphes idéaux décrivant les N classes de symboles, la stratégie adoptée consiste à générer un ensemble de NxK prototypes de graphes intégrant de possibles distorsions dans les graphes. Ces prototypes sont ensuite utilisés dans la phase de classification pour déterminer la classe de symboles inconnus et bruités.

L'approche proposée peut être décomposée en 3 étapes. Dans un premier temps, un algorithme de génération de bruit est appliqué sur l'image de chaque symbole modèle afin d'obtenir un ensemble de M images de symbole par classe. Le bruit appliqué, fréquemment utilisé dans la communauté de la reconnaissance de symboles, a été reconnu comme représentatif du bruit qui intervient dans les images de documents [VAL 04]. A partir des MxN images obtenues, l'algorithme d'apprentissage est appliqué. Il repose sur un processus d'optimisation dont l'objectif est de générer l'ensemble des K graphes prototypes par classe. L'algorithme d'optimisation choisi est un algorithme génétique que nous avons adapté à la manipulation des graphes. Le critère à optimiser est le taux de reconnaissance que permet d'obtenir l'utilisation des prototypes dans une simulation de classification par Plus Proche Voisin (1-PPV). Cette simulation de classification est appliquée sur une base de test distincte de la base d'apprentissage. Pour effectuer cette phase de classification par 1-PPV, il est évidemment nécessaire de définir une distance dans l'espace des paramètres, donc ici dans l'espace des graphes. Pour ce faire, l'approche choisie repose sur une mesure de dissimilarité calculée à partir de l'extraction de signature de graphes (*graph probing)*. Finalement, une phase de classification basée sur cette même mesure est appliquée sur la base de validation pour évaluer la qualité des prototypes générés.

La suite de l'article est organisée de la façon suivante. Dans la seconde section, le concept de signature de graphes est présenté. Puis, l'algorithme génétique dédié à l'évolution des graphes est exposé dans la section 3. La section 4 propose quant à elle l'application de l'approche proposée à la reconnaissance de symboles. Une comparaison de différentes mesures de dissimilarité entre graphes ainsi que les résultats de reconnaissance obtenus y sont présentés. Enfin, une conclusion sur ce travail est dressée et les perspectives sont décrites.

2 Mesures de dissimilarité entre graphes

Les mesures de dissimilarité entre objets complexes structurés (ensembles, listes, chaînes, graphes…) sont généralement basées sur la quantité de termes communs. La mesure la plus simple est appelée coefficient de ressemblance ; elle consiste à calculer le nombre d'objets communs. Dans le cas des graphes, une telle mesure consiste alors à déterminer le plus grand sous-graphe commun [BUN 98] entre les graphes pour évaluer leur ressemblance. Une autre approche existante pour mesurer la dissimilarité entre deux graphes est la distance d'édition. Elle repose sur le calcul d'un coût de transformation pour passer d'un graphe à l'autre. Une séquence d'opérations d'édition (insertion, suppression, modification…) permettant le passage d'un graphe à l'autre doit d'abord être déterminée. Puis, le coût de cette séquence est calculé en sommant le coût de chacune des opérations unitaires. Dans [BUN 97], il est prouvé que la distance d'édition de graphes et la détermination du plus grand sous-graphe commun sont équivalentes pour un ensemble particulier de valeurs de coûts. On peut également montrer que les deux mesures ont des complexités algorithmiques exponentielles dans le pire des cas.

Dans le cadre de l'approche proposée dans l'introduction, nous avons exposé le fait qu'un algorithme génétique est utilisé pour le calcul de prototypes de graphes. Une telle approche requiert de nombreux appels à la mesure de dissimilarité entre graphes. C'est pourquoi il est nécessaire d'adopter une approche rapide pour cette mesure, au détriment de sa précision. Dans un tel contexte, la topologie du graphe peut alors être partiellement ignorée en considérant une approximation basée uniquement sur l'ensemble des nœuds ou des arcs. Deux approches ont été proposées dans [KRI 03]. Une fonction de coût de mise en correspondance de deux arcs permet de calculer une distance entre deux graphes par une recherche de coût minimal pour la plus grande mise en correspondance possible entre les ensembles d'arcs des deux graphes. Dans le pire des cas, la complexité de l'algorithme correspondant est en $O(n3)$, où n est le nombre d'arcs du plus grand des deux graphes.

Une autre possibilité pour qualifier la proximité de deux graphes est de décrire chacun des graphes par un vecteur numérique et de calculer ensuite de façon classique la distance entre ces deux vecteurs. La technique dite des signatures de graphes décrite dans [LOP3] consiste à construire le vecteur numérique en dénombrant au sein du graphe les occurrences de sous-graphes particuliers nommés empreintes. Cette technique s'exécute en temps linéaire, moyennant la recherche des empreintes. Cette recherche est particulièrement rapide lorsque les graphes empreintes sont de taille réduite. Toutefois, une distance nulle entre les vecteurs numériques décrivant les graphes n'implique pas l'isomorphisme des graphes comparés. Il ne s'agit donc pas d'une distance entre graphes, mais plutôt d'une mesure de dissimilarité. Cependant, dans leur article, Lopresti et Wilfong ont démontré une relation permettant de minorer la distance d'édition par rapport à la mesure de dissimilarité basée sur les empreintes. De ce fait, cette mesure de dissimilarité est un bon indice de la proximité effective des graphes. Son temps de calcul la rend adaptée à son utilisation par un algorithme génétique. La section 4 présente une étude expérimentale comparant cette mesure avec d'autres approches.

3 Un Algorithme génétique opérant sur des graphes

Les algorithmes génétiques (AG) sont des algorithmes d'optimisation évolutionnaires inspirés de l'évolution Darwinienne des populations. Ils réalisent une exploration parallèle de l'espace des paramètres en exploitant les résultats précédemment obtenus par les individus (solutions possibles) des générations précédentes. Cette exploration est en partie aléatoire mais reste toutefois guidée par le but à atteindre. La figure 1 donne les grandes phases d'un algorithme génétique générique ainsi que les étapes nécessaires à son application à un problème spécifique. Après une initialisation aléatoire d'une population de solutions possibles, quatre opérations sont enchaînées : la sélection, la reproduction, le croisement et la mutation. Ces opérations se fondent sur trois étapes préalablement réalisées par l'utilisateur : la définition du codage des solutions, la définition de la fonction objectif et l'éventuelle définition d'opérateurs dédiés au problème. Dans la littérature, le codage des solutions est généralement linéaire, les individus sont des chaînes de valeurs réelles ou binaires. Les opérateurs, dont le but est de permettre l'échange de matériel génétique entre les solutions de bonne qualité sont alors l'échange de gênes autour de points choisis aléatoirement (généralement 1 ou 2) pour le croisement, et la modification ponctuelle de la valeur d'un petit ensemble de gênes pour la mutation.

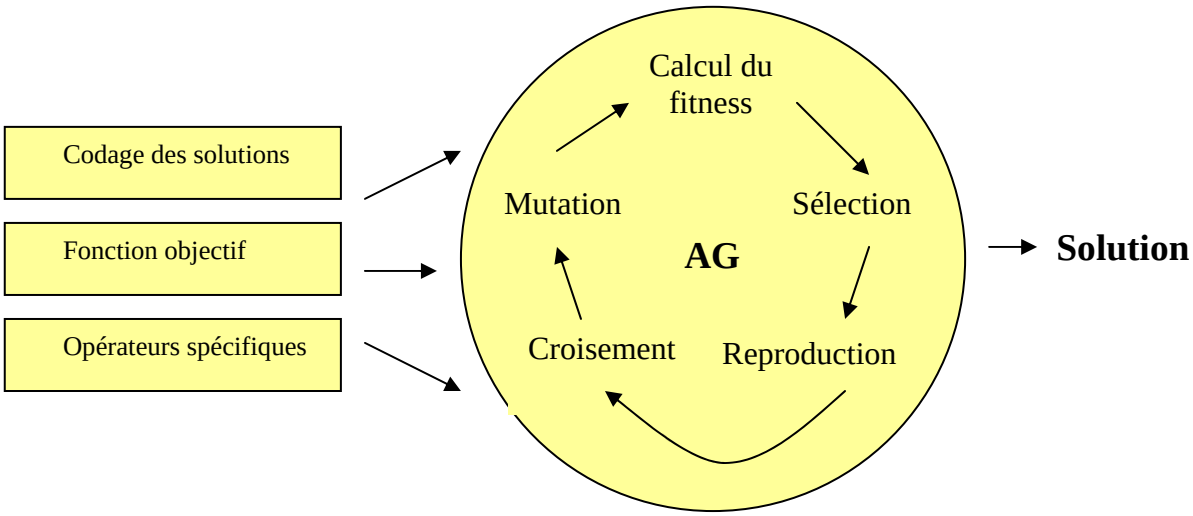

FIG 1: Vue d'ensemble d'un algorithme génétique

Dans notre contexte de reconnaissance structurelle de symboles par sélection de prototypes, chaque individu de la population représente un ensemble donné de graphes (les *K*x*N* prototypes choisis). L'évolution de tels individus au cours des générations implique donc de revisiter les opérateurs génétiques classiques pour que ceux-ci permettent la modification de structures de type graphes. Pour la mutation, l'opérateur utilisé par l'algorithme proposé est naturel. Il se base sur les six opérations unitaires qui peuvent être appliqués à un graphe : ajout ou suppression d'un nœud, ajout ou suppression d'un arc, et modification de l'étiquette d'un nœud ou d'un arc. A chaque application d'une opération de mutation, un tirage aléatoire permet de décider laquelle des opérations doit être appliquée, ainsi que la nouvelle valeur d'étiquette le cas échéant. Concernant le croisement (Figure 2), une séparation aléatoire de l'ensemble des nœuds en deux sous-ensembles est d'abord réalisée. Les arcs sont alors étiquetés, soit en arc interne si les nœuds extrémités sont dans le même sous-ensemble, soit en arc externe si l'arc joint deux nœuds de deux sous-ensembles distincts. Ensuite, les nœuds sont eux aussi classifiés en nœuds de sortie s'ils sont à la source d'un arc externe ou en nœuds d'entrée s'ils sont la destination d'un arc externe. Cette classification permet alors de permuter les sous-ensembles en fonction de la classe des nœuds. Les arcs sont re-combinés de façon à ce que les arcs externes pointent vers des nœuds d'entrée choisis aléatoirement. Les deux opérations de croisement et de mutation sont appliquées séquentiellement, après un processus classique de sélection basé sur l'algorithme de la roue de loterie.

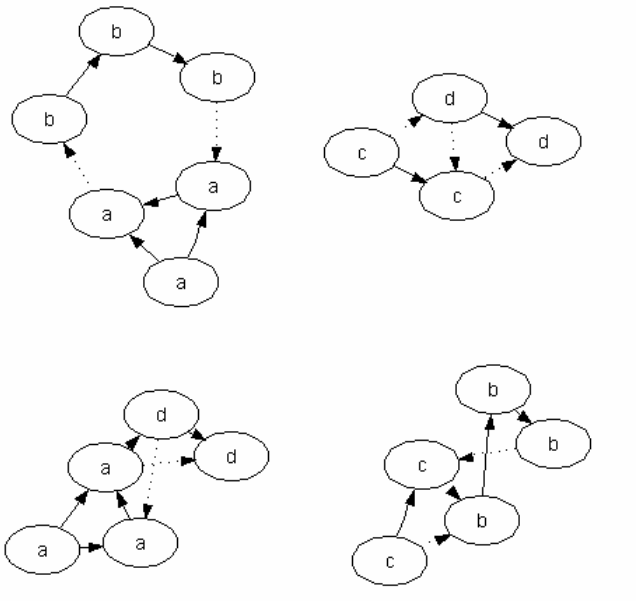

FIG 2: L'opérateur de croisement sur les graphes

4 Application

Cette section présente comment les concepts décrits précédemment sont appliqués à notre problème de reconnaissance de symboles. Nous présentons d'abord l'étape de construction de graphe pour illustrer ensuite une étude comparative des mesures de dissimilarité. Cette étude permet de justifier notre choix porté vers l'utilisation de signatures de graphes et de choisir les valeurs de certains paramètres de la représentation en graphes. Finalement, des résultats liés à notre application de reconnaissance de symboles sont présentés et comparés à d'autres approches.

4.1 Construction de la base de graphes

Les données que nous manipulons sont des graphes extraits d'un corpus de 180 images de symboles et générés à partir de 10 modèles idéaux de symboles provenant de la base GREC (Graphic RECognition Workshop symbol recognition contest). Des composantes connexes (blanches et noires) sont tout d'abord extraites des images binaires des symboles pour être automatiquement étiquetées à l'aide d'un algorithme de classification non supervisée [KAU90] utilisant les moments de Zernike comme caractéristiques [KHO 90]. Cet algorithme permet de trouver un nombre *c* de regroupements. Ensuite, et en considérant un nombre maximal *h* de voisins significatifs pour chaque composante ainsi étiquetée, nous construisons un graphe dont les nœuds correspondent à ces composantes. Deux nœuds sont reliés par un arc si l'un des deux appartient aux *h* proches voisins de l'autre. Les valeurs de *c* et de *h* ont été fixées à partir de l'étude comparative entre les mesures de dissimilarité décrite dans la partie suivante. La figure 3 montre quels sont les graphes construits pour deux images de symboles.

4.2 Comparaison entre les mesures de dissimilarité

Dans le but de choisir la meilleure mesure de dissimilarité dans le contexte de notre application, nous avons mené une étude des valeurs de corrélation entre les différentes mesures proposées dans la section 2. Une première expérience dans cette étude utilise les coefficients de corrélation de Pearson (cor) entre les différentes mesures de dissimilarité dont nous disposons. Les résultats sont présentés sur la première ligne du tableau 1. La seconde expérience considère la corrélation entre un ordre de vérité terrain défini par un utilisateur (ordre total ou partiel) et l'ordre calculé à l'aide d'une distance entre représentations de graphe. Dans ce cadre, au vu de notre objectif de classification de graphes, c'est l'utilisateur qui détermine suivant ses propres critères l'ordre de ressemblance des symboles, et on cherche à ce que la corrélation entre les mesures de dissimilarité et cet ordre soit la plus élevée possible. Cette corrélation peut être mesurée en utilisant le coefficient (tau) de corrélation de rang défini par Kendall [KEN 55]. Grâce aux valeurs obtenues, nous pouvons alors sélectionner une représentation de graphe (c'est à dire effectuer le choix des valeurs des *c* et *h* définis dans la partie précédente) et une mesure de dissimilarité qui satisfassent à la fois les contraintes de complexité et de temps de traitement, et une très grande corrélation avec la vérité terrain définie pour notre application. Les résultats obtenus en termes de corrélation (tableau 1) combinés aux complexités respectives des approches font apparaître la mesure basée sur les signatures de graphes comme étant le meilleur compromis au vu de nos contraintes de temps de calcul.

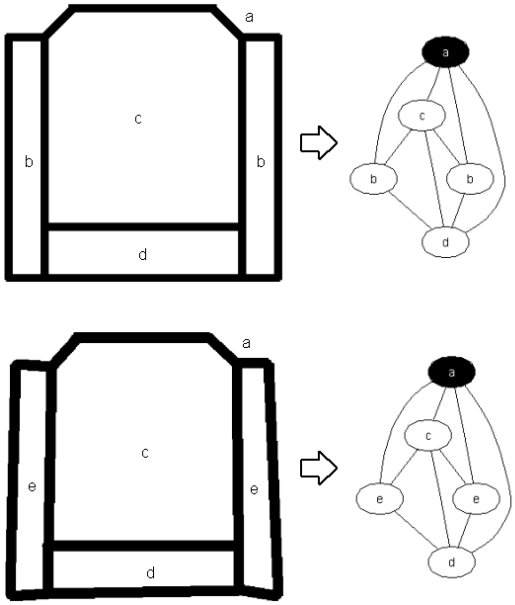

FIG 3: Construction des graphes

	DE	SG	DK
DE avec cor	1	0.58	0.63
DE avec tau	1	0.53	0.63
VT avec tau	0.699	0,622	0,657

TAB 1. Mesures de Corrélation entre la distance d'édition (DE), la Distance de Kriegel (DK), les signatures de graphes (SG) et la vérité terrain (VT)

4.3 Résultats de reconnaissance

Nous détaillons dans cette section les approches d'apprentissage et de classification utilisées ainsi que les résultats de reconnaissance obtenus. Comme nous l'avons déjà indiqué, l'algorithme d'apprentissage consiste à générer *K* prototypes de graphe pour chacune des *N* classes de symbole dont nous disposons. Ces prototypes sont obtenus grâce à l'utilisation d'Algorithmes Génétiques dont le but est de trouver la solution quasi-optimale à un problème de reconnaissance qui utilise ces prototypes. Dans un tel contexte, chaque individu de notre population est un vecteur contenant K graphes par classe, c'est-à-dire *K* solutions possibles (prototypes) pour chaque classe. Ainsi, un individu est

composé de *K*x*N* graphes. A l'initialisation de la population initiale, chaque graphe de chaque individu est construit aléatoirement à partir du corpus de graphes initial. Le taux de « fitness » de chaque individu est alors mesuré grâce au taux de classification obtenu pour une base de test et les prototypes correspondants. L'approche de classification utilise le classifieur du 1-PPV et le principe des signatures de graphes. L'Algorithme Génétique itère alors en utilisant les opérateurs décrits dans la section 3 avec l'objectif d'optimiser le taux de classification. Le critère d'arrêt est ici simplement le nombre de générations effectuées. Une étape de classification est alors appliquée sur une base de validation pour évaluer la qualité des prototypes sélectionnés. Nous pouvons finalement comparer les résultats obtenus avec une approche qui, elle-aussi, recherche K meilleurs représentants dans un ensemble d'objets décrits par une matrice de distances réciproques. Cette approche minimise les distances des objets à leurs représentants, il s'agit de l'approche PAM (Partition Around Mediods) [10]. PAM permet donc d'obtenir les K meilleurs représentants pour chacune de nos classes. A l'aide de ces prototypes obtenus et de la distance issue du calcul de signatures, nous obtenons alors des taux de reconnaissance que le tableau 2 compare pour des valeurs de *K*=1,2 ou 3 prototypes par classe, et pour un ensemble de 10 classes. Pour compléter ce tableau, notons que si seul le modèle idéal est utilisé en tant qu'élément apprentissage, le classifieur du 1-PPV n'offre qu'un taux de reconnaissance de 88%. Tous ces résultats montrent donc l'intérêt de la sélection de prototypes à l'aide d'Algorithme Génétique combiné avec une approche de signatures de graphes. En effet l'apprentissage permet de créer *N*x*K* éléments synthétiques grâce aux opérateurs génétiques dans l'objectif d'obtenir la meilleure représentation d'une classe, et l'étendue des possibilités n'est pas limitée au(x) graphe(s) constituant une classe.

K	1	2	3
PAM	81,14%	95,22%	96,66%
GA	98,92%	99,17%	99,44%

TAB 2: Taux de reconnaissance obtenus

5 Conclusion

Dans cet article, nous avons proposé un algorithme de classification de graphes appliqué à un problème de reconnaissance de symboles. L'approche est basée sur l'apprentissage de prototypes de graphes à l'aide d'algorithmes génétiques. Le concept de signature de graphe est utilisé pour mesurer la dissimilarité entre graphes. Ce choix a été fait suite à une étude comparative entre différentes mesures qui a montré le bon compromis rapidité / efficacité qu'elle permettait d'obtenir. Les résultats obtenus par le système complet ont montré l'intérêt de la génération de prototypes synthétiques par l'utilisation d'opérateurs génétiques plutôt que l'utilisation des éléments de la classe de symboles.

Les perspectives à ce travail concernent différents points. La première concerne l'enrichissement de la description des symboles en intégrant les résultats de la vectorisation des formes. Une seconde concerne la validation de l'approche sur une base de données plus importante. Ce travail va être effectué en utilisant les données proposées par le projet Technovision EPEIRES.

A plus long terme, nous envisageons la comparaison de l'approche proposée avec une approche à base de SVM s'appuyant sur des noyaux de graphes. Enfin, la transformation de l'approche proposée pour lui intégrer une possibilité de rejet est également considérée. Dans ce cadre, il sera nécessaire d'utiliser un autre algorithme d'optimisation capable d'appréhender plusieurs objectifs.

6 Bibliographie

[LLA 01] LLADOS J., VALVENY E., SANCHEZ G. ET MARTI E., Symbol recognition: Current advances and perspectives, *Lecture Notes in Computer Science*, N° 2390, 2001, pp 104-127.

[SCH 03] SCHENKER A., LAST M., BUNKE H. ET KANDEL A.. Classification of web documents using a graph model, *Actes de la 7ème International Conference on Document Analysis and Recognition (ICDAR)*, 2003, pp 240-244.

[SER 05] SERRAU A., MARCIALIS G.L., BUNKE H. ET ROLI F., "An experimental comparison of fingerprint classification methods using graphs", *Lecture Notes in Computer Science*, N° 3434, 2005, pp 281-290

[VAL 04] VALVENY E. ET DOSCH P., Symbol Recognition Contest: A Synthesis". *Lecture Notes in Computer Science*, N° 3088, 2004, pp. 368-385.

[BUN 98] BUNKE H. ET SHEARER K., A graph distance metric based on the maximal common subgraph, *Pattern Recogn. Lett.*, 19, 1998, pp 255-259.

[BUN 97] BUNKE H., On a relation between graph edit distance and maximum common subgraph", *Pattern Recogn. Lett.*, 18, 1997, pp 689-694.

[KRI 03] KRIEGEL H.P. ET SCHÖNAUER S., Similarity Search in Structured Data, *Lecture Notes in Computer Science*, N° 2737, 2003, pp. 309-319.

[LOP 03] LOPRESTI D. P. ET WILFONG G.T., A fast technique for comparing graph representations with applications to performance evaluation, *International Journal on Document Analysis and Recognition*, 6, 2003, pp 219-229.

[KEN 55] KENDALL M. G., Rank Correlation Methods, *Hafner Publishing Co*., New York, 1955.

[KAU 90] KAUFMAN L., ET ROUSSEEUW P.J., Finding groups in data, *John Wiley & Sons*, Inc., New York, 1990

[KHO 90] Khotazad A. et Hong Y.H., Invariant image recognition by Zernike Moments, *IEEE PAMI*, Vol 12, No 5, May 1990, pp 489-497.