Multilabel-Chain-Verfahren

Idee:

Das Multilabel-Chain-Verfahren ist ein speziell für die Multilabel-Klassifikation entworfenes Verfahren, das ähnlich wie das Classifier-Chain-Verfahren arbeitet. Das Besondere an dem Multilabel-Chain-Verfahren ist, dass zufällige Entscheidungsbäume verwendet werden, die nicht an eine bestimmte Lernaufgabe angepasst sind. Dies bietet den Vorteil, dass keine Klassifikations-Reihenfolge der Labels vorgegeben werden muss, sondern diese direkt bei der Klassifikation dynamisch bestimmt wird. In den Entscheidungsbäumen werden hierfür Tests verwendet, die auf Label-Attributen testen. Wie bei den zufälligen Entscheidungsbäumen wird auch hier das zu testende Label-Attribut zufällig ausgewählt. Hierbei kann jeder Entscheidungsbaum keine, einen oder mehrere beliebige Label-Attribute überprüfen. Über einen Parameter wird eingestellt, wie viele Label-Tests das Ensemble von Entscheidungsbäumen enthalten soll.

Um eine Instanz zu klassifizieren, werden alle Entscheidungsbäume des Ensembles mehrmals durchgegangen. In der ersten Iteration haben wir noch kein Label vorhergesagt und deswegen müssen alle Label-Tests in den Bäumen übersprungen werden. Bei dem Überspringen der Label-Tests wird die Instanz an alle Kindknoten weitergeleitet und die Vorhersagen der Kindknoten werden anschließend kombiniert. Mit der erhaltenen Vorhersage des Ensembles kann dann entschieden werden, welches Label-Attribut vorhergesagt werden soll. Nachdem ein der Wert für ein Label-Attribut vorhergesagt wurde, wird der Prozess wiederholt, mit der Änderung, dass nun die Label-Tests, die auf dem vorher vorhergesagten Attribut testen, mitverwendet werden können. In jedem Durchgang wird genau ein Label vorhergesagt, welches in der darauffolgenden Iteration zur Erstellung der Vorhersage verwendet werden kann. Dies wird solange wiederholt, bis alle Labels vorhergesagt wurden.

Um den Entwicklungsprozess zu vereinfachen, soll das Verfahren erstmal nur für das **BATCH-Lernen** entworfen werden.

Die initiale Form des Lernalgorithmus hat folgende Parameter:

Name des Parameters	Beschreibung
numTrees	Anzahl der Bäume im Ensemble
maxDeep	Maximale Tiefe der Entscheidungsbäume
maxS	Minimale Anzahl an Instanzen in einem Knoten, um einen Test zu erstellen (Zum Beispiel, wenn an einen Knoten zehn Instanzen weitergeleitet werden und maxS den Wert 15 hat, dann wird anstelle eines inneren Knotens ein Blatt erstellt).
percentageLabels	Der prozentuale Anteil an Tests die ein Label-Attribut überprüfen (Ist beispielsweise der Wert 10%, dann überprüfen ungefähr 10% von allen Tests in dem Ensemble Label-Attribute)
predictCombination	Die Art, wann das Label-Attribut vorhergesagt wird: 1. PRE_CHAIN: Bei diesem Verfahren erstellt jeder Entscheidungsbaum in dem Ensemble eine Vorhersage. Bei jeder Vorhersage wird entschieden, welches Label-Attribut vorhergesagt werden soll. Anschließend wird das Label-Attribut vorhergesagt, das die meistens Votes erhalten hat.

	2. POST_CHAIN: Bei diesem Verfahren erstellt jeder Entscheidungsbaum in dem Ensemble eine Vorhersage, jedoch werden diese Vorhersagen zuerst kombiniert. Anhand der kombinierten Vorhersage wird dann ein Label ausgewählt, das vorhergesagt werden soll. Bei der Kombination der Vorhersagen der Bäume werden die Vorhersagen normalisiert, so dass jeder Baum den gleichen Einfluss auf die kombinierte Vorhersage hat.
predictType	Die Art, wie das nächste zu vorhersagende Label in Kombination mit dem „chainType“ ausgewählt wird: 1. PERCENTAG: Hierbei wird während des Lernprozesses die Häufigkeit eines Labels gespeichert. Ist ein Label in sechs von zehn Instanzen „true“ in einem Blatt, dann entspricht das einem Wert von 60% für das Label. Das Besondere bei PERCENTAGE ist, dass ein Label in einer Iteration nur mit „true“ vorhergesagt werden kann, wenn die Häufigkeit größer oder gleich als 50% ist. Wenn ein Label unter 50% Häufigkeit hat, kann es nur mit „false“ vorhergesagt werden. 2. LABEL: Bei diesem Verfahren wird neben der Häufigkeit der Labels auch der Mittelwert der Anzahl an Labels in den Trainingsinstanzen gespeichert (Regression über Anzahl der Labels). Bei der Vorhersage müssen idealerweise immer genau so viele Labels den Wert „true“ haben, wie auch der Mittelwert der Labels ist (hierbei wird auf oder abgerundet). Dies ermöglicht es auch Labels vorherzusagen, die beispielsweise nur eine Häufigkeit von 40% haben. Hat ein Label ein Vorkommnis von größer oder gleich 50% und die Anzahl der mit „true“ vorhergesagten Labels ist kleiner als der Mittelwert der vorhergesagten Labels, dann kann dieses Label mit „true“ vorhergesagt werden. Müssen wir nur noch ein Label vorhersagen und die Anzahl der vorhergesagten Labels ist kleiner als der Mittelwert der Anzahl der Labels, dann wird das Label mit „true“ vorhergesagt, egal wie hoch das Vorkommnis ist.
chainType	Die Art, wie das nächste zu vorhersagende Label in Kombination mit dem „predictType“ ausgewählt wird: 1. LOWEST: Es wird versucht zuerst alle Labels mit dem Wert „false“ vorherzusagen. 2. HIGHEST: Es wird versucht zuerst alle Labels mit dem Wert „true“ vorherzusagen. 3. HYBRID: Es wird das Label vorhergesagt, bei dem man sich am sichersten ist, dass es „true“ oder „false“ ist. Diese Sicherheit wird über das Vorkommnis der Labels bestimmt. Um ein Label mit „true“ zu klassifizieren muss die Vorkommnis nahe an 100% sein und um ein Label mit „false“ zu klassifizieren muss die Vorkommnis nahe an 0% liegen. Je nachdem welches Label den geringsten Abstand zu den entsprechenden Wert hat, wird vorhergesagt.

Erste Treffen:

Die ersten Versuche werden analysiert. Hierbei hatte ich auch ein paar Visualisierungen zu dem Treffen mitgebracht, konnte bis jetzt nur einen Teil finden (ich kann mich aber noch erinnern). Das Hauptaugenmerk lag hierbei auf den Parametern „chainType“ und „predictCombination“. An denen Versuchen ließ sich eindeutig erkennen, dass die Methoden „HIGHEST“ und „LOWEST“ nicht so gute Ergebnisse hervorbringen wie „HYBRID“ (Ergebnisse sind im Ordner „results_HYBRID_HIGHEST_LOWEST“), und dass das „PRE_CHAIN“ Verfahren auch deutlichere schlechtere Ergebnisse produziert, als das „POST_CHAIN“ Verfahren. Wenn man darüber genauer nachdenkt, macht das auch Sinn. In dem weiteren Verlauf der Entwicklung des Multilabel-Chain-Verfahrens werden diese schlechteren Methoden nicht mehr verwendet.

Darüber hinaus wird vorgeschlagen zwei neue Parameter dem Verfahren hinzuzufügen, mit denen das Multilabel-Chain-Verfahren genauer untersucht werden kann. Zum einen gibt es den Parameter „numChain“, mit welchem sich einstellen lässt, wie viele Labels mit dem Verfahren vorhergesagt werden sollen. Ist beispielsweise der Wert fünf, so werden die ersten fünf Labels durch das Multilabel-Chain-Verfahren vorhergesagt und anschließend in der nächsten Iteration alle restlichen Labels mit der resultierenden Vorhersage ermittelt. Hiermit kann die Auswirkung der iterativen Vorhersage analysiert werden. Der andere Parameter, welcher hinzugefügt wird, ist „percentageActiveLabels“, welcher angibt wie viele der vorhandenen Label-Tests in dem Ensemble verwendet werden sollen. Hierzu können die Label-Tests den Status aktiv und inaktiv haben. Inaktive Label-Tests werden immer übersprungen (egal ob der Wert für das Label schon gesetzt wurde) und aktive Label-Tests funktionieren wie normale Label-Tests (wenn der Wert für das Label vorhanden ist, kann die Instanz an den entsprechenden Kindknoten weitergeleitet werden.)

Folgerungen + Nächste Schritte:

- Als „chainType“ wird ab nun nur noch „HYBRID“ verwendet
- Als „predictCombination“ wird nur noch „POST_CHAIN“ verwendet
- Hinzufügen des Parameters „numChain“
- Hinzufügen des Parameters „percentageActiveLabels“
- Durchführen von Versuchen mit den neu entworfenen Parametern (hierbei soll die Auswirkung der Variation der neu entworfenen Parameter analysiert werden)

Die weiterentwickelte Form des Lernalgorithmus hat folgende Parameter:

Name des Parameters	Beschreibung
numTrees	Anzahl der Bäume im Ensemble
maxDeep	Maximale Tiefe der Entscheidungsbäume
maxS	Minimale Anzahl an Instanzen in einem Knoten, um einen Test zu erstellen (Zum Beispiel, wenn an einen Knoten zehn Instanzen weitergeleitet werden und maxS den Wert 15 hat, dann wird anstelle eines inneren Knotens ein Blatt erstellt).
percentageLabels	Der prozentuale Anteil an Tests die ein Label-Attribut überprüfen (Ist beispielsweise der Wert 10%, dann überprüfen ungefähr 10% von allen Tests in dem Ensemble Label-Attribute)
predictType	Die Art, wie das nächste zu vorhersagende Label in Kombination mit dem „chainType“ ausgewählt wird: 1. PERCENTAGE: Hierbei wird während des Lernprozesses die Häufigkeit eines Labels gespeichert. Ist ein Label in sechs von zehn Instanzen „true“ in einem Blatt, dann entspricht das einem Wert von 60% für das Label. Das Besondere bei PERCENTAGE ist, dass ein Label in einer Iteration nur mit „true“ vorhergesagt werden kann, wenn die Häufigkeit größer oder gleich als 50% ist. Wenn ein Label unter 50% Häufigkeit hat, kann es nur mit „false“ vorhergesagt werden. 2. LABEL: Bei diesem Verfahren wird neben der Häufigkeit der Labels auch der Mittelwert der Anzahl an Labels in den Trainingsinstanzen gespeichert (Regression über Anzahl der Labels). Bei der Vorhersage müssen idealerweise immer genau so viele Labels den Wert „true“ haben, wie auch der Mittelwert der Labels ist (hierbei wird auf oder abgerundet). Dies ermöglicht es auch Labels vorherzusagen, die beispielsweise nur eine Häufigkeit von 40% haben. Hat ein Label ein Vorkommnis von größer oder gleich 50% und die Anzahl der mit „true“ vorhergesagten Labels ist kleiner als der Mittelwert der vorhergesagten Labels, dann kann dieses Label mit „true“ vorhergesagt werden. Müssen wir nur noch ein Label vorhersagen und die Anzahl der vorhergesagten Labels ist kleiner als der Mittelwert der Anzahl der Labels, dann wird das Label mit „true“ vorhergesagt, egal wie hoch das Vorkommnis ist.
percentageActiveLabels	Parameter, der nur für das Durchführen von Experimenten existiert. Gibt an, wie viele der vorhandenen Label-Tests in dem Ensemble verwendet werden sollen. Hierzu können die Label-Tests den Status aktiv und inaktiv haben. Inaktive Label-Tests werden immer übersprungen (egal ob der Wert für das Label schon gesetzt wurde) und aktive Label-Tests funktionieren wie normale Label-Tests (wenn der Wert für das Label vorhanden ist, kann die Instanz an den entsprechenden Kindknoten weitergeleitet werden.)
numChain	Parameter, der nur für das Durchführen von Experimenten existiert. Anzahl an Labels, die mit dem Multilabel-Chain-Verfahren vorhergesagt werden sollen. Ist beispielsweise der Wert fünf, so werden die ersten fünf Labels durch das Verfahren vorhergesagt und anschließend in der nächsten Iteration alle restlichen Labels mit der resultierenden Vorhersage ermittelt. Hiermit kann die Auswirkung der iterativen Vorhersage analysiert werden

Anschließende Mails:

Es wurden zwei Arten von Experimenten durchgeführt. Das erste Experiment variiert den Parameter „percentageActiveLabels“ und das zweite Experiment variiert den Parameter „numChain“. Die Ergebnisse wurde in Koordinatensystem visualisiert, wobei auf der x-Achse der variable Parameter und auf der y-Achse der Wert für das entsprechende Evaluierungsmaß abgebildet ist. Beide Versuche wurden mit einem Ensemble von 30 Bäumen("trees") mit einer maximalen Tiefe von 15("deep") und 10%, 20% und 30% Label-Tests("percentageLabels") durchgeführt.

In dem Versuch mit „percentageActiveLabels“ ließ sich folgende Beobachtungen machen:

- Je mehr Label-Tests aktiv sind, desto besser sind die Ergebnisse. Bei dem Datensatz "Emotions" kann man sogar bei der PERCENTAGE-Methode einen Anstieg der Subset-Accuracy von 2% bis auf 30% erkennen. Ähnliches Verhalten bei der Subset-Accuracy ist deutlich auch bei den Datensätzen "Yeast", "Flags" und ("medical") zu erkennen. Etwas weniger deutlich ist es bei den anderen Datensätzen.

Den Versuch „numChain“ haben wir nicht wirklich untersucht, aber hier lassen sich die gleichen Beobachtungen erkennen:

- Je mehr Labels durch das Multilabel-Chain-Verfahren hervorgesagt werden, desto besser sind die Ergebnisse

Allgemein lässt sich erkennen:

- Die Text-Datensätze lassen sich nicht gut klassifizieren (liegt eventuell an der geringen Tiefe der Bäume). Die Ergebnisse der Evaluierungsmaße unterscheiden sich stark von den State-of-the-Art-Methoden.

Folgerungen + Nächste Schritte:

- Es sollen einige Baselines berechnet werden. BR und CC mit Basisklassifizierer RDT mit 30 Bäumen und einer Tiefe von 15. Darüber hinaus soll auch der J48 als Basisklassifizierer mit initialen Parametereinstellungen verwendet werden.

Die Grafiken der Evaluation sind in dem Ordner „results_PERCENTAGEACTIVE_NUMCHAIN“ beigelegt. Bei beiden Versuchen habe ich die Ergebnisse auf zwei Weisen dargestellt, als "overview_datasets" und als "overview_measures".

- *Bei "overview_datasets" habe ich die Ergebnisse der "Post-Chain"-Methoden ("Percentage" und "Label") für 10%, 20% und 30% Label-Tests in einem Plot dargestellt. Für jeden Datensatz ist ein PDF vorhanden, in dem alle Ergebnisse der Evaluationsmetriken enthalten sind.*
- *Bei "overview_measures" wird für jede Evaluationsmetrik und jede Parametereinstellung (hier nur unterschiedlich durch "percentageLabels" = 10%, 20% und 30%) ein PDF erzeugt. In dieser PDF sind für jeden Datensatz die Ergebnisse dieser Evaluationsmetrik dargestellt. In diesen Plots sind vier Linien eingezeichnet: "Label-Normal" ist die normale Vorhersage mit der Regression der Anzahl der zu vorhersagenden Labels; "Percentage-Normal" ist die normale Vorhersage, bei der alle Labels vorhergesagt werden, die öfter als 50% mal gesehen wurden. Bei "Percentage-Post-Chain" und "Label-Post-Chain" handelt es sich um die CC-Methoden.*

Zweites Treffen + folgende Mails:

Auf die Ergebnisse der Baslines wird nur teilweise eingegangen (die Parametereinstellungen mit 30 Bäumen und einer Tiefe von 15 ist auch nicht wirklich repräsentativ/aussagekräftig). Bei dem Treffen überwiegt die Erkenntnis, dass das Verfahren schlecht auf den Text-Datensätzen funktioniert.

Des Weiteren wird beschlossen neue Versuche für das Multilabel-Chain-Verfahren durchzuführen. Hierbei soll eine Parameteroptimierung auf dem Trainings-Datensatz durchgeführt werden. Anschließend sollen die optimalen Parametereinstellungen verwendet werden, um die Test-Set Evaluation durchzuführen. Dabei sollen die folgenden Methoden verwendet werden:

- RDT mit Label-Kollektor
- RDT mit Percentage-Kollektor (BR-Methode)
- RDT-Chain (Multilabel-Chain-Label)
- RDT-Chain (Multilabel-Chain-Percentage)
- RDT-Chain (mit 50 zufälligen Chain-Orderings)
- RDT mit Label-Powerset-Kollektor
- J48 mit Label-Powerset (keine Parameteroptimierung)
- J48 mit Binary Relevance (keine Parameteroptimierung)
- J48 mit Classifier-Chain (mit 50 zufälligen Chain-Orderings) (keine Parameteroptimierung)

Für die Optimierungen werden alle Kombinationen der folgenden Parameter evaluiert:

- trees = {500}
- maxDeep = {5,10,15,20,25,30,35,40}
- maxSs = {2,5,8}
- percentageLabels = {0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5}

Bei dem Datensatz "CAL500" ist leider kein Trainings-Set und Test-Set vorhanden. Das Trainings-Set besteht nun aus den ersten 350 Instanzen und das Test-Set besteht aus den restlichen 150 Instanzen.

Für die Principle Component Analyse wird auch eine Art Optimierung mit folgenden Parametereinstellungen durchgeführt:

- valuesR = {1.0, 0.5}
- valuesN = {10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 200, 300, 400, 500, 600, 700, 800, 900, 1000}

Jeden erstellten PCA-Datensatz habe ich mit einer 20-fachen Kreuzvalidierung auf den Trainingsdaten evaluiert. Für die Evaluation habe ich den normalen RDT mit Label-Kollektor verwendet. Hierbei wurden nur folgende, provisorische Parametereinstellung verwendet:

- trees = 500
- deep = 20
- maxS = 2

Am Ende wird der PCA-Datensatz ausgewählt, der die besten Ergebnisse erzielt hat. Dieser wird dann für die Optimierung aller Algorithmen verwendet.

Folgerungen + Nächste Schritte:

- Anwendung der PCA-Analyse auf den Text-Datensätzen.
- Versuche mit Parameteroptimierung durchführen.

Folgende Mails:

Bei der Optimierung der Parameter der PCA-Analyse lassen sich Verbesserungen mit dem verwendeten RDT-Algorithmus (Label-Kollektor, trees=500, deep=20, maxS=2) erkennen.

Datensatz	Subset-Accuracy ohne PCA	Subset-Accuracy mit PCA
Enron	1,98%	5,7%
Medical	16.52%	35,44%

Diese Ergebnisse kommen ebenfalls jedoch nicht die State-of-the-Art-Methoden ran, jedoch wurde auch noch keine Parameteroptimierung auf den resultierenden PCA-Datensätzen durchgeführt. Die PCA-Optimierung konnte insgesamt für die Datensätze „Enron“, „Medical“ und „Corel5k“ durchgeführt werden.

Ergebnisse der Evaluation (mit den optimalen Parametern, die durch die Parameteroptimierung gefunden wurden)

1. Die Ergebnisse der RDT-Verfahren des Datensatzes "Flags" sind im Vergleich zu den Ergebnissen der Optimierung sehr schlecht (Optimierung: Subset-Accuracy RDT-Chain-Percentage = 26.36%; Test-Set: Subset-Accuracy RDT-Chain-Percentage = 16.31%). Das liegt wahrscheinlich daran, dass für die Optimierung nur sehr wenige Instanzen verwendet werden konnten.
2. Genau das umgekehrte Verhalten zeigt sich bei dem Datensatz "Scene". Hier wurde für RDT-Label bei der Optimierung nur 50% Subset-Accuracy erreicht, hingegen bei der Test-Set-Evaluation 70% Subset-Accuracy erreicht werden. Das liegt wahrscheinlich an der vordefinierten Verteilung der Instanzen auf Test- und Trainings-Set.
3. Allgemein hat sich herausgestellt, dass der Label-Powerset-Kollektor in Verbindung mit RDTs sehr gute Ergebnisse erzielt (siehe Datensätze "Scene", "Emotions" und "Enron").
4. Die entworfenen RDT-Chain-Verfahren können leider nur auf den Datensätzen "Yeast", "Emotions" und ("Enron") überzeugen.
5. An den Ergebnissen lässt sich auch erkennen, dass nur die PCA-Transformation von dem Datensatz „Enron“ gute Ergebnisse erzielt. Die Ergebnisse der anderen PCA-Transformationen sind schlecht.

Measure: Subset Accuracy									
	RDT-Label	RDT-Percentage	RDT-LabelPowerset	RDT-Chain-Label	RDT-Chain-Percentage	RDT-Chain-Given	J48-BR	J48-CC	J48-LP
Flags	0.1385	0.1231	0.1538	0.1385	0.1538	0.1846	0.0769	0.1846	0.2
Scene	0.7124	0.1622	0.7099	0.4983	0.4975	0.2667	0.4013	0.4958	0.536
Yeast	0.1778	0.0829	0.1876	0.2072	0.2279	0.1658	0.0643	0.1276	0.1167
Emotions	0.198	0.2178	0.3663	0.2129	0.3317	0.2327	0.1287	0.2475	0.2178
CAL500	0	0	0	0	0	0	0	0	0
enron (PCA)	0.0363	0.0777	0.1589	0.0674	0.1416	NOT GIVEN	0.0933	0.114	0.1157
medical (PCA)	0.3519	0.1085	0.369	0.3643	0.0946	NOT GIVEN	0.0016	0.014	0.0109
Corel5k (PCA)	0.002	0	0	0	0	NOT GIVEN	0.01	0.016	0.018
Mediamill	0.024	0.0786	NOT GIVEN	0.0451	0.1095	NOT GIVEN	0.0535	0.0806	0.0718
Birds	0.356	0.483	0.4892	0.4025	0.4799	0.4799	0.4861	0.483	0.4675

Ergebnisse der Subset-Accuracy nach der Test-Set Evaluation, die mit den optimierten Parametern durchgeführt wurde. Die komplette Datei mit den Resultaten ist ebenfalls beigelegt („results_TEST_SET“).

Drittes Treffen:

Die Ergebnisse der Test-Set Evaluation werden nochmals analysiert. Man kommt zum Schluss, dass PCA nicht so gut funktioniert und, dass die Label-Power-Set Variante zu gut ist. Des Weiteren wird darüber gesprochen, wie die Klassifikation abläuft. Hierbei fällt auf, dass die Vorhersagen der Kindknoten, wenn ein Label-Test übersprungen wird, gemergt (normalisiert zusammengefügt) werden. Dies wirft die Frage auf, ob das einfach addieren der Vorhersagen bessere Ergebnisse hervorbringt. Bei dem Problem mit den Sparse-Datensätzen ist man erstmal ratlos. Die Ergebnisse der Kreuzvalidierung und der Test-Set Evaluierung bei dem Datensatz Scene unterscheiden sich stark.

Folgerungen + Nächste Schritte:

- Durchführen der Parameteroptimierung und der anschließenden Test-Set Evaluation mit den optimalen Parametern (Diesmal werden die Vorhersagen beim Überspringen eines Label-Tests addiert). Dies soll dann mit der Variante „merge“ verglichen werden.
- Gedanken über das Sparse-Problem machen.
- Der Datensatz „Scene“ hat eine komische Aufteilung der Instanzen in Trainings- und Test-Set, weswegen bei der Test-Set Evaluierung meistens deutlich bessere Ergebnisse herauskommen. (Dies haben wir im Nachhinein auch belegt, indem wir die optimalen Parametereinstellungen mit der Test-Set Evaluierung und mit einer 10-fachen Kreuzvalidierung laufen gelassen haben. Hierbei kam raus, dass ein deutlicher Unterschied existiert. Bei der Verwendung des Label-Kollektors mit dem normalen RDTs kamen wir zu folgenden Ergebnissen bei gleichen Parametereinstellungen: CV=46,31% Subset-Accuracy und Test-Set 71,24% Subset-Accuracy)

Folgende Mails:

Analysieren der Ergebnisse bei den die Vorhersagen beim Überspringen von Label-Tests mit „ADD“ oder mit „MERGE“ kombiniert werden. Hierbei wurde für die Kombinationsart "ADD" nochmal die ganze Parameteroptimierung durchgeführt und die entsprechenden Parametereinstellungen bei der Test-Set-Evaluierung verwendet. In dem Ordner "results_ADD_MERGE" befinden sich die Ergebnisse der TestSet-Evaluation.

In der folgenden Abbildung sind die Ergebnisse „MERGE“ dargestellt (gemittelt nach 5 Wiederholungen)

Measure: Subset Accuracy						
	Flags	Scene	Yeast	Emotions	CAL500	Birds
RDT-Label	0.1292 +/- 0.0157	0.7079 +/- 0.0038	0.1751 +/- 0.0050	0.2010 +/- 0.0120	0.0000 +/- 0.0000	0.3523 +/- 0.0066
RDT-Percentage	0.1508 +/- 0.0329	0.1569 +/- 0.0028	0.0875 +/- 0.0033	0.2129 +/- 0.0077	0.0000 +/- 0.0000	0.4824 +/- 0.0030
RDT-LabelPowerset	0.1508 +/- 0.0062	0.7177 +/- 0.0043	0.1974 +/- 0.0065	0.3703 +/- 0.0037	0.0000 +/- 0.0000	0.4966 +/- 0.0042
RDT-Chain-Label	0.1538 +/- 0.0138	0.5194 +/- 0.0156	0.2146 +/- 0.0061	0.2356 +/- 0.0170	0.0000 +/- 0.0000	0.3783 +/- 0.0086
RDT-Chain-Percentage	0.1692 +/- 0.0097	0.5189 +/- 0.0160	0.2238 +/- 0.0058	0.3485 +/- 0.0161	0.0000 +/- 0.0000	0.4799 +/- 0.0000

In der folgenden Abbildung sind die Ergebnisse „ADD“ dargestellt (gemittelt nach 5 Wiederholungen)

Measure: Subset Accuracy						
	Flags	Scene	Yeast	Emotions	CAL500	Birds
RDT-Label	0.1292 +/- 0.0157	0.7079 +/- 0.0038	0.1751 +/- 0.0050	0.2010 +/- 0.0120	0.0000 +/- 0.0000	0.3517 +/- 0.0077
RDT-Percentage	0.1508 +/- 0.0329	0.1569 +/- 0.0028	0.0875 +/- 0.0033	0.2129 +/- 0.0077	0.0000 +/- 0.0000	0.4824 +/- 0.0030
RDT-LabelPowerset	0.1508 +/- 0.0062	0.7177 +/- 0.0043	0.1974 +/- 0.0065	0.3703 +/- 0.0037	0.0000 +/- 0.0000	0.4966 +/- 0.0042
RDT-Chain-Label	0.1508 +/- 0.0062	0.7079 +/- 0.0062	0.2150 +/- 0.0054	0.2495 +/- 0.0102	0.0000 +/- 0.0000	0.4947 +/- 0.0046
RDT-Chain-Percentage	0.1600 +/- 0.0075	0.6259 +/- 0.0126	0.2236 +/- 0.0075	0.3485 +/- 0.0080	0.0000 +/- 0.0000	0.4799 +/- 0.0000

Allgemein lässt sich in den Ergebnissen von "ADD" und "MERGE" nur deutliche Unterschiede bei den Datensätzen "Scene" und "Birds" erkennen. Durch das Verfahren "RDT-Chain-Label" werden bei "ADD" deutlich bessere Ergebnisse erzielt. Bei dem Datensatz "Scene" erreichen wir sogar die 70% Subset-Accuracy, die bei "RDT-Label" auch erreicht werden. Mit "RDT-Chain-Percentage" kann mit "ADD" bei diesem Datensatz eine ungefähr 10% bessere Subset-Accuracy erreicht werden. Bei dem Datensatz "Birds" haben wir nur für das Verfahren "RDT-Chain-Label" eine ungefähr 10% bessere Subset-Accuracy. (Werte der Subset-Accuracy befinden sich auf der nächsten Seite, leider konnten nur die Datensätze Flags, Scene, Yeast, Emotions, CAL500 und Birds evaluiert werden.)

Folgerung + Nächste Schritte:

- Das Kombinieren der Vorhersagen mit „ADD“ erzielt bessere Ergebnisse und wird ab sofort nur noch verwendet.
- Das Problem mit den Text-Datensätzen ist noch nicht gelöst.

Lange Zeit geschieht nichts.

Mail im August + Treffen im September:

Lösungsansatz für das Problem mit den Text-Datensätzen: Ein neuer Splitter wurde entwickelt, der besonders gut mit sparse Daten arbeiten. Hierbei wird zufällig eine Instanz aus den verfügbaren Instanzen ausgewählt und alle Attribute, die den Wert 1 haben, werden notiert. Aus diesen Attributen wird dann zufällig ein Attribut ausgewählt, das dann in den Splitter verwendet wird. Das funktioniert relativ gut, da das Problem vorher war, dass wir sehr oft Attribute in einem Splitter getestet hatten, bei den alle Instanzen nur an einen Kindknoten weitergereicht wurden, also keine Informationen gewonnen wurden. Dieses neue Verfahren wird auch für die Auswahl des Attributs für die Label-Tests angewendet, was den Parameter „percentageLabels“ etwas sensibler macht.

Auf den Text-Datensätzen "Enron" und "Medical" wurde ein kleines Experiment durchgeführt, bei dem ich das alte Verfahren mit dem neuen Verfahren vergleiche. Die Ergebnisse (Grafiken) sind in dem Ordner „results_SPARSE“ beigelegt. Hier lässt sich eine deutlich bessere Performanz mit dem Multilabel-Chain-Verfahren auf den Sparse-Datensätzen erkennen, welche den State-of-the-Art-Methoden aus dem Rule-Paper entsprechen oder besser sind.

Des Weiteren glaube ich, dass man durch die Entwickelung einer neuen Art zur Multilabel-Klassifikation auch noch ein bisschen die Performance verbessern kann (also einen neuen Kollektor bzw. ein neues Chain-Verfahren entwickeln). Deutlich kann man das an den Ergebnissen von den Datensätzen „Enron“ und „Medical“ erkennen. Auf „Enron“ ist PERCENTAGE sehr gut und auf „Medical“ ist LABEL sehr gut. Der LABEL-Kollektor und der PERCENTAGE-Kollektor haben nämlicher weise beide gewisse Vorteile und Nachteile. Darüber hinaus habe ich mir ein paar Gedanken über ein LabelPowerset-Chain-Verfahren gemacht, hierbei bin ich mir aber noch nicht sicher, wie das Verfahren die Labels am besten vorhersagen soll.

Folgerungen + Nächste Schritte:

- Der Ansatz für das Problem mit den Text-Datensätzen wird weiterentwickelt, wobei Daniel diese Arbeit wahrscheinlich übernehmen wird.
- Es könnte die Art der Chain-Methoden überarbeitet werden (Kombination aus LABEL und PERCENTAGE)
- Es könnte ein Label-Power-Set-Chain-Verfahren entwickelt werden.
- Das Sparse-Splitter könnte auch bei den anderen Datensätzen für die Label-Attribute verwendet werden.
- Mit dem neuen Verfahren könnten die Parameteroptimierungen und die anschließende Test-Set Evaluierung wiederholt werden.