NLP and Sentiment Driven Automated Trading

Atish Davda (adavda@seas.upenn.edu)

Parshant Mittal (pmittal@seas.upenn.edu)

Faculty Advisor: Michael Kearns (mkearns@cis.upenn.edu)

Abstract

Movements in financial markets are directly influenced by information exchange – between a company and its owners, between the government and its citizens, between one individual and another. The channels of distributing news have expanded from the singular ticker tape in the middle of town to intra-minute delivery to the computer via RSS feeds. With information quickly available markets are becoming increasingly efficient, as humans design intricate algorithms to continuously take advantage of any perceived mispricing in the markets (Kelly, 2007). This phenomenon, which is especially prevalent in the stock market, begs the question: is there still an active need for the human element? After all, machines are faster – given more information and better hardware, their computation power decidedly exceeds that of humans. The answer lies in the challenge of abstraction; deciding the impact of each piece of information is important and more isn't always better (Greenwald, Jennings, & Stone, 2003) .

In this project we explored the field of natural language processing and identified methods we can use to automate stock trading based on news articles. The project was implemented in three phases (see Appendix 1). The first phase included data collection from sources on the web. News articles and headlines were scraped from Yahoo! Finance; historical market data was collected from Google Finance. The data was collected for 600 small market cap stocks (SML), 400 medium market cap stocks (MID) and 500 stocks from S&P 500 index (SP500). The second phase included sentiment analysis on the first half of the dataset, in order to compute sentiments to be tested on the (out of sample) second half. In the final stage, we implemented an NLP approach to quantifying the headlines. This was done using a number of NLP packages available online, including the Stanford Lex Parser, WordNET, and General Inquirer.[1] The last stage of the project comprised of developing a trading module with which we could incorporate the results of historical market, sentiment, and NLP analysis to give a *Buy*, *Sell*, or a *Hold*

[1] Please refer to the *Bibliography* section for further information on these projects.

recommendation for securities under consideration. Using sentiment and NLP analysis we were able to achieve significantly improved returns. In fact we averaged a return of 4.0% over a two month period (27% annualized), while the market fell 8.7% during the same period (-42.1% annualized). With the help of this and other metrics, we explored the value of NLP in automated trading.

Related Work

Given the widespread implications of introducing abstraction capability to machines, it isn't surprising that NLP is a highly researched discipline. In fact, even in just the US there exist several groups sponsored by universities, corporations, and the government, which focus solely on improving the capabilities of current language-processing techniques (Fallows, 2004). However, although the paradigm of examining news articles attracts a lot of academic studies, it is rather biased toward long-term, *macro* news reports[2]; unexplored by comparison, is the realm of short-term, firm-specific news.[3] One of the first studies specifically focused on quantifying the relationship between news releases and movements in the stock markets was conducted not too long ago (Gillam, Ahmad, & Ahmad, 2002).

The challenge of predicting which news events will have what impact on the trading characteristics, such as price and volume traded of stocks still remains. While there have been recent advancements in the applications of NLP in predicting other markets (e.g. election markets), the specific role of language analysis in financial markets is unclear (Gilder & Lerman, 2007). The novelty of our project lies in applying NLP analysis to news headlines, rather than the entire article. In addition, we consider highly liquid and efficient markets. These markets present additional challenges as there is no end date and our analysis must then include a wider range of factors. One natural dimension we explored in detail was distinguishing the impact between the headline "IBM's earnings drop" and "IBM's earnings

[2] Macro news reports include interest rate changes by central banks, announcements of inflation news, etc.
[3] Firm-specific news includes earnings reports, merger/acquisition rumors, etc.

plummet."[4] Our paper is, in part, an extension of the 2002 study "Economic News and Stock Market Correlations" which solely looked at the *sign* (positive or negative) of the connotation associated with words in the news articles. We have implemented a framework with the use of General Inquirer as well as our own sentiment analysis to distinguish between the emotional *charges* people innately give to certain words, which lead to varying degrees of influence the news has on the characteristics of the stock. Upon additional research on generic topics such as conjunctive handling, we found a good fit for such fundamental pillars of NLP (Meena & Prabhakar, 2007). Furthermore, we expanded upon this kind of study by examining syntax in addition to semantics, empirically deriving an adjustment factor to each word's sentiment charge, depending on its use in a sentence. This second order correction helped improve accuracy of predictions, once we moved away from the naïve bag-of-words analysis.

Building this lexicon with each word having an associated sentiment is a field of research in itself, Sentiment Analysis. There are several models for generating such a corpus; one of the fundamental models is described in the paper "Determining the Sentiment of Opinions" by Kim and Hovy (2004). The study discusses a *region* (news headline) around the central *anchor* (company of interest), which when examined as a whole, yields a positive or negative rating for the company itself (Kim & Hovy, 2004). Another approach suggests a more empirical analysis by examining vast amounts of HTML documents in order to generate a *polarity* score for words, described as a function of the *distance* of a given word from a pre-defined, manually selected corpus (Kaji & Kitsuregawa, 2007). While it would certainly help having a sentiment list as close to perfect as possible, we focused on the *use* of sentiment scores, rather than determining the optimal method to calculate them. As you will read in the *Technical Approach* section, we adopted a combination of these two methodologies along with General Inquirer – initially, we used a discretionary method akin to the latter model, and eventually, will develop a hybrid.

[4] Gilliam article titled "Economic News and Stock Market Correlation" discusses the impact of "good" versus "bad" words, but does not incorporate degrees of positive/negative sentiment associated with the word.

The project's goal is two-fold: one is to test whether a relationship exists between news articles and the movements in the market data of a stock; the second goal is to model this relationship, if it exists, by implementing it into a trading strategy. In regard to the former, the scope of news content can be broadly divided into two sets: news reporting on past performance, and announcements of future activity (Gillam et al., 2002). While it would be an interesting dimension to explore, this study limits itself to quantifying relationships between characteristics of news articles and relevant stock returns, *regardless* of the category under which the news falls. The reason for doing so is because we focus on implementing this strategy as if it were to be used in a high frequency event driven trading platform where it is often acceptable to be accurate just little over 50%. The reason hinges on consistently being right over half the time, so that the profits generated will more than account for the losses sustained due to incorrect decisions. Detailed analysis on the subject of Statistical Arbitrage has been performed, by testing various experimental trading strategies used to test predictive effects of news releases on stock movement (Hariharan, 2004).

While Hariharan's ideas are in a way predecessors to the space of stock trading based on news release, this project delves more into the realm of NLP in the context of financial textual data, rather than the development of a trading strategy (which is a secondary focus of the project).[5] Primarily, the project will explore and attempt to derive a predictive relationship between news reports and stock movements. Another study by Subramanian, aimed at optimization of automated trading algorithms would have come in handy in later phases, had we decided to focus on strategies. Rather, we employed a *simple* set of trading ideas, described later, to quantify and avoid confounding the results with advanced models (Subramanian, 2004).

[5] If we happen to make significant progress towards our goals of achieving satisfactory NLP accuracy, we may begin to shift our focus on refining the trading strategy tailored to the results.

An interesting source of data encountered during the preliminary research stages was the TextMap Suite of web-accessible services (Skiena, 2007). The suite of tools provides visual representations of vast amounts of potentially helpful time-series data on various topics. While we can fathom several uses of this data, we did not make much use of it. Although it is out of the scope of this project, a very interesting extension to our design would be a crawler module, discussed further in the Future Work section.

Although this project strictly focused on the aforementioned ideas, we tried to build a platform, which we could foresee being extended. In particular, what distinguished this project from some of its predecessors is that it maintained a continuous log of parsing output, correlations computations, and trading decisions. We implemented this functionality with the vision that possibly an extension could be developed which analyzes calculations and decisions made by the system, in both the NLP and trading stages, to determine the *source* of incorrect guesses. By doing so, the enriched predictive model could help increase the accuracy of the predictions by computing correlations between more tightly related data sets.[6]

In order to measure and compare the results, one obvious measure to consider is the correlation computed based on the news vector and the stock movements.[7] While it may be tempting, as we discuss later, calculating correlations adds little outside providing some qualitative intuition; by itself, it is not a sufficient measure of the relationship between sentiments and stock movements (this is where trading strategies are required). That said, as you see in Appendix 2, we inferred relationships from correlations and were able to verify them with a trading strategy (see *Trading Results*). The goals are to

[6] An example of the improvement in the predictive model would be to compute appropriate weights for various factors (repetition of phrases, mention of CEO, etc), which ultimately play a role in calculating the correlation between news data and stock movements.
[7] Please read about some of our metrics in the *Technical Approach* section.

continuously improve performance, as established by certain metrics, to determine how effectively the system proves or disproves our conjecture, and details on performance evaluation are outlined in the *Technical Approach* section that follows.

Technical Approach

The project was finished using a three phase system.[8] The reason for using a phase system was so that we would have something concrete after each manageable phase. Not to mention, it is easier to modify the design on the fly rather than pre-specifying under-researched modules. In Phase One of our project, we built a comprehensive data collection model. The analysis relies very heavily on this and as a result, utmost care was taken to make sure the data we were using was accurate. Perl scripts were used to extract news headline data from Yahoo! Finance. Most of this data was in HTML which was parsed and inserted into a database. We initially employed MS Excel to import data from Bloomberg. However dealing with some 150,000 price data points each with multiple attributes proved to be slow and tedious process. As a result we soon switched over to Java and MySQL. Our Java program scraped Google Finance for historical market data and after careful pruning the raw data; this dataset was inserted in a MySQL database. Using MySQL, rather than flat files, made the processing much faster and easier to deal with data in Java. Using this information we were able to run naïve market strategies, which became our base-metric, from which we would try to improve applying sentiment and NLP analyses.

After data collection, we focused on building a sentiment analyzer. We had originally planned to use General Inquirer (GI) to build the initial word list, however we were not able to acquire license necessary to use it. We did manage to get our hands on some of the raw data that GI uses. This dataset was then

[8] See Appendix 1 for a detailed schematic of the three phases.

set up on our MySQL to allow for some functionality that resembled GI's. Nonetheless, in order to calibrate our sentiments appropriately, we soon realized we needed a lot of data. Therefore, although we had only acquired three months worth of data (160,000+ headlines) we modified our scripts to back-fetch another three months' data. This afforded us a larger "learning bed" for our sentiment-analyses. We thus decided to build our own sentiment analyzer. Using the first half of our dataset we computed sentiments for words on two parameters, frequency of the word in our dataset as well as the stock return associated with the headlines containing the words. The words were then normalized and assigned a score of +10 to 0 for positive words and 0 to -10 for negative words. Our original lexicon had hand-selected 200 high-frequency words. These words were then "stemmed," or reduced to their basic indefinite verb form to ensure uniqueness, using the Stanford Lex Parser's stemming algorithm. Using historical returns as a proxy for sentiment, we reverse engineered sentiments from stock price movements. A corpus of 200 words, however, seemed too small as we did not see an equitable distribution of positive and negative verbs. We then employed WordNET's synonym tagged dictionary, and ultimately expanded the list to over 2100 words. These expanded words received a sentiment as a function of weighted averages of words from the original list, whose synonyms they are. For instance, if the word *advance* is received from WordNET as a synonym of the words *progress* and *forward*, *advance* would be given a sentiment score as an average of the scores of *progress* and *forward*. This expansion ensured a 40-60 distribution of positive-negative sentiment words in our 2100+ word corpus. We believe it would have been difficult to muster a 50-50 mix given that **all our analysis and tests were performed during July-November 2007** – a time of extremely bearish outlooks on equity financial markets.

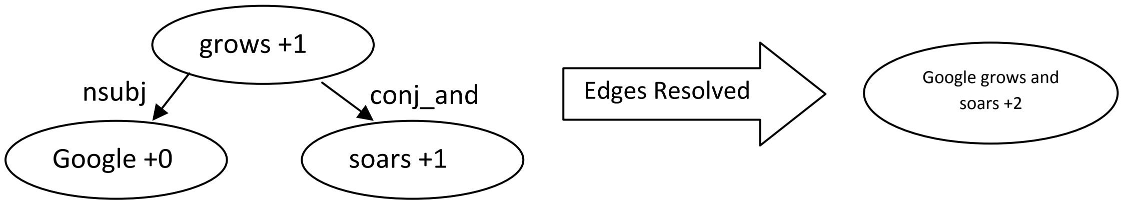
Multiple such lists were created, each with a different list of words and sentiments, *some of which* include: the smallest manual 200-word list to those from General Inquirer; some lists were created including or excluding the synonyms words (sizes 2100 and 1500 respectively). After calculating sentiments for each word, a sentiment was calculated by for each headline as the simple sum of the sentiment of each of its constituent words. This bag-of-words analysis was a modest attempt at sentiment analysis before we applied more complex NLP grammar rules. More involved attempts to abstract away meaning from headlines are discussed below. For each attempt at calculating a headline sentiment (including bag-of-words) the individual headline sentiments were aggregated by corresponding stock for each day, giving each stock a “score” for each day (thus, enabling us to give a *Buy*, *Sell*, or *Hold* opinion). One thing to note regarding our dataset is that the use of sentiment analysis for headlines made quite a bit of sense. Since headlines are often fragments instead of proper sentences, they do not conform to proper grammatical structure. It is therefore difficult to perform full-fledged sentence-level (even phrase level) analysis on them. In such a situation sentiment analysis can (and did) improve results. Although the bag-of-words analysis was extremely crude, as you will see in the *Trading Results* section, it seemed to improve our results significantly.

In the third phase of the project, we focused on NLP analysis of the headlines. We wanted to understand the ideas expressed in the headline by examining the interrelationships of the words in the headline. To do so, we first extract the relationship between the words using the Stanford Parser which given a headline will output an interrelationship between words. For example, it will pick out the verb and the subject and highlight that the one word is the subject of the words. The relationships that Stanford Parser detects are listed in Appendix 3, with relationships to the right being more specific. Based on such information one can make a decision regarding the headlines. We took these relationships and create a tree rooted on the most important word (often the main verb in the sentence). In the tree, the vertex was the word and the edge was the relationship between those words. The vertex also contained other information regarding the sentiment expressed by the tree root at that vertex. The analysis of the entire tree was done by starting at the leaf nodes and then percolating up until we reached the root. In order to move from a child node to its parent, the edge connecting them had to be resolved. Resolving this edge allowed us to assign a quantifiable entity which could then be used to express a view on the stock. To resolve the edge we developed rules based on the relationship between the child and the node. The complete list of relationships that were in our tree is in Appendix 3. One of the important rules in the set are the governing conjunctions. For example, if we had the headline "*Google grows and soars",*

The raw relationships between the words are expressed as:

```
nsubj(grows-2, Goolge-1)
conj_and(grows-2, soars-4)
```

Or in our tree form as



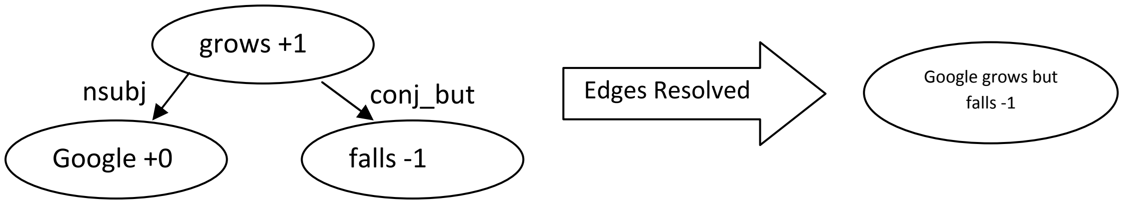
The relationship between "grows" and "soars" is connected by an '*and'* and we resolve it by adding the quantifiable score of soars to growth's so that the two ideas summed can amplify the positive effect on Google.

In the case the headline was *"Goolge grows but falls"*

The raw relationships between the words are expressed as:

```
nsubj(grows-2, Goolge-1)
conj_but(grows-2, falls-4)
```

Or in our tree form as



The relationship between “grows” and ”falls” is connected by a ‘*but’* and we resolve such an edge by taking the score of the child (the word following ‘but’) in the tree.

We developed rules for other relationships as well. The rules were developed by using a sample of the headlines and seeing what ideas were expressed by the relationships. Additionally we also viewed the grammar and usage rules regarding these relationships in order to form our rules. This kind of approach has had some success in the past. The best example that we came across was the one by Meena and Prabhakar (Meena & Prabhakar, 2007) who built their trees around conjunctions and applied conjunction rules to derive a value for the tree. We also employed conjunction rules, however we decided root our trees around verbs as we felt verbs express the main idea of the headline.

The NLP trees were also evaluated in multiple flavors. The differences in evaluation resulted from which corpus (expanded, intra-index, General Inquirer, etc) we decided to use. Having developed multiple methods of qualifying the headlines we were then ready to test our hypotheses. We decided to build a flexible trading module, with capabilities for using different decision making criterion (see *Trading Results*). From these results, we see the basic spectrum of success (especially when considered net of the declining market) of these various strategies.

Trading Results

We spent a great deal of time trying to conjecture which types of trading strategies (criteria for *buy*/*sell*/*hold*) would be most profitable. Based on the very preliminary correlations we had noticed (Appendix 2), we had a few guesses: we posited that a strategy involving the volume data should be profitable. Similarly, we decided to try a news count (for stock per given day) based model. The issues we faced (and realized) only once we began assigning headlines a sentiment was: even for a simplistic bag-of-words analysis, which list of sentiments do we use? Sentiments aggregated over all indices, only over the respective index, General Inquirer sentiments, etc. These issues only compounded once we introduced the aforementioned NLP analysis. We decided to strike a balance between a biased ruling out of any strategy, and making educated guesses as to which would have an intuitive explanation.

You can see our choice of strategies in Appendix 5, where we have listed the *Name* of each strategy along with its description. This *Name* will be your key in deciphering the graphs that follow in Appendix 6. The reason we tested a fair number of strategies, is that we had economical or financial justifications for each one, but wanted to determine which method of analyzing headlines was more superior. Our hypotheses (eleven possible strategies, and one baseline: the market itself) were mainly a result of trying to run a conjoint-type study to see which method of determining sentiment, and which NLP (dependency tree) method was superior. Remember, we varied our sentiment scoring by using only the sentiment dataset developed from the headlines in the stock's corresponding index as well as one sentiment dataset based on the global headline dataset. For all practical purposes we did not notice much of a difference between these two. Additionally another choice we made was to calculate returns as a close-price on previous day to close-price today (as opposed to the difference between open and close on the same day). After reading some literature on the matter, we decided that we as the investors bear the most risk when there is most time for news to be released without ability to take action; that is, between the close of one day and the open of the next day (Koppel & Shtrimberg, 2006). For this reason, we decided to use with a close-close return. The returns were calculated as a generic equal-weighted portfolio on each index. Specifically, take the example of the mid-index. For *every stock* comprising the mid-index data, we *give* $1 as the initial investment in that stock. This money grows or shrinks based on the performance of the respective stock. The total of these mini-investments yields a portfolio, called MID, which allows us to calculate on any day what the to-date cumulative return is of the strategy used to make the buy/sell decisions. In Appendix 6 we see exactly a time-series of these returns for different strategies. Notice the losses the market (PLTS_50) takes, while other strategies incrementally improve as investment opportunities. The legend for the different strategies is in Appendix 5. Please refer to the table in Appendix 7 to see the relative successes of each strategy. As you can see, the Bag-of-Words, the News Count, and the General-Inquirer/Sentiment strategies are very close (statistically insignificant) and thus, the *winners*. In the appendix, you will see various strategies we tested, and you will notice that surprisingly, the Bag-Of-Words analysis is almost as good (at times if not better) than the dependency tree (NLP) method. This suggests, not the inferiority of the NLP method, but rather the fact that different jobs call for different tools – for headlines, which are not real sentences, dependency trees don't fare so much better than the simple bag-of-words as we would expect. It is to infer nuances like these that we mention the inadequacies of calculating a correlation between sentiment and a stock's return. By designing an explicit trading model with rules of when to buy, when to sell, one can truly isolate the effect of different approaches of NLP, better than just calculating a one-number summary. Furthermore, while our trading models in this project (all eleven variations) were fairly simple, more complicated models (like the last model in Appendix 5 – a potential improvement) can be used where we maintain sequential statistics, running means and standard deviations to refine our decision-making. As we mentioned in *Related Work* we have decided to keep a log of all decisions made by the trading strategy, and its result, so that if possible, there can be some regressions or analysis performed, to determine which corner cases the strategy is failing to capture.

Technical Challenges

Over the course of the project we ran into number of difficulties. For one, we had quite a bit of data that we needed to search quickly. At first we were hesitant to use a database only because neither of us had much experience which with it. But after the initial learning curve, we think that was one of the best decisions we made. Our data was much more reliable, it was flexible and mostly importantly it was faster than our other choices. Additionally, the two of us are often working on different parts of the system and sometimes there is miscommunication as to how exactly one component should work. This results in wasted time as well nightmares in debugging. Furthermore as our code grew, it become harder and to manage to files and integrate. One approach we are taking to decrease the time we spend on such issues is to have a good follow the good software engineering rules. Furthermore, as we calculated statistics and gathered data, the amount of data grew tremendously, and every time we performed a calculation it took longer and longer to find the values needed. We were able to mitigate some of these types of problems by using MySQL and have it be optimized for the type of calculations we were doing.

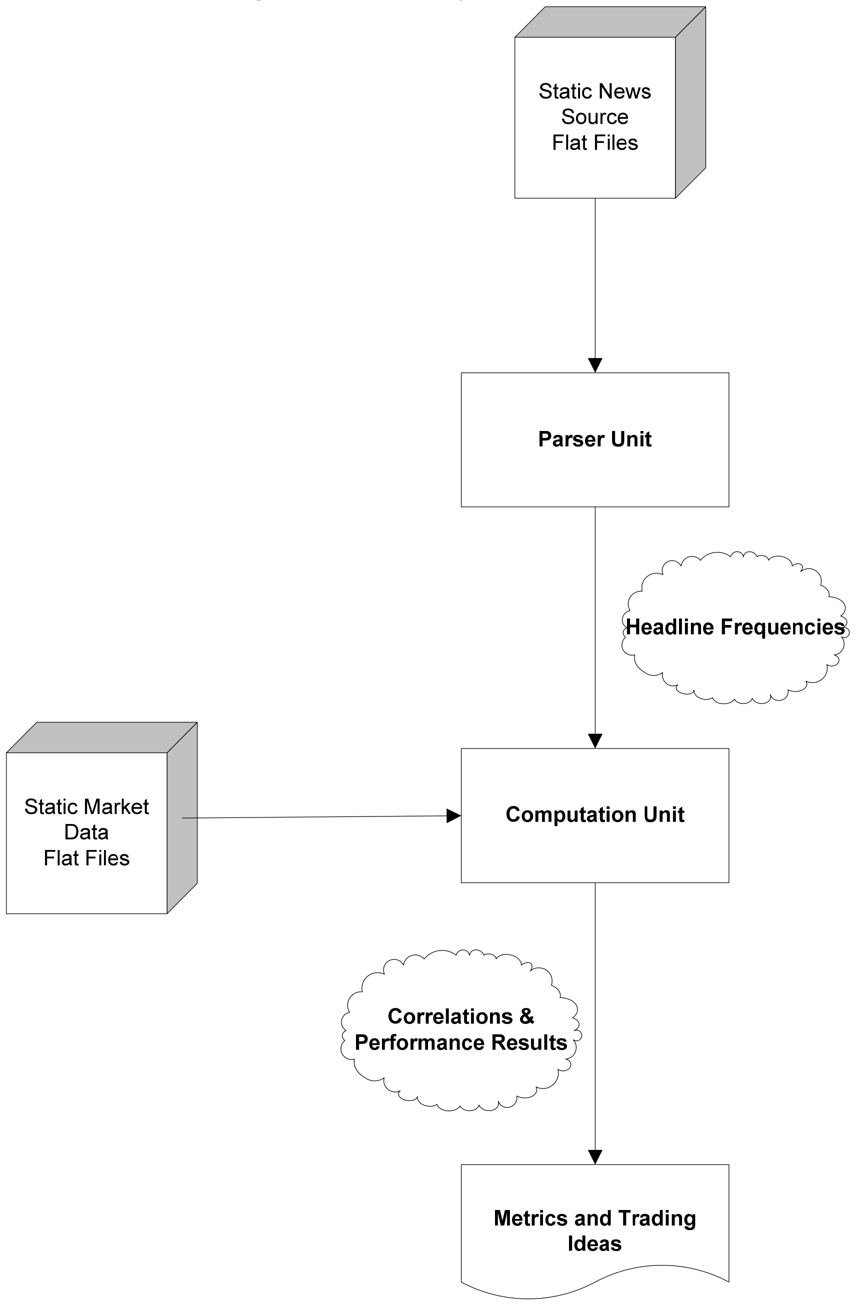
Yet another technical/logistical difficulty we are experienced was getting General Inquirer. However we were lucky in that we were able to get the raw data that GI uses and then set it up locally so we could go through the data and extract the information that GI would have provided to us.

Future Work

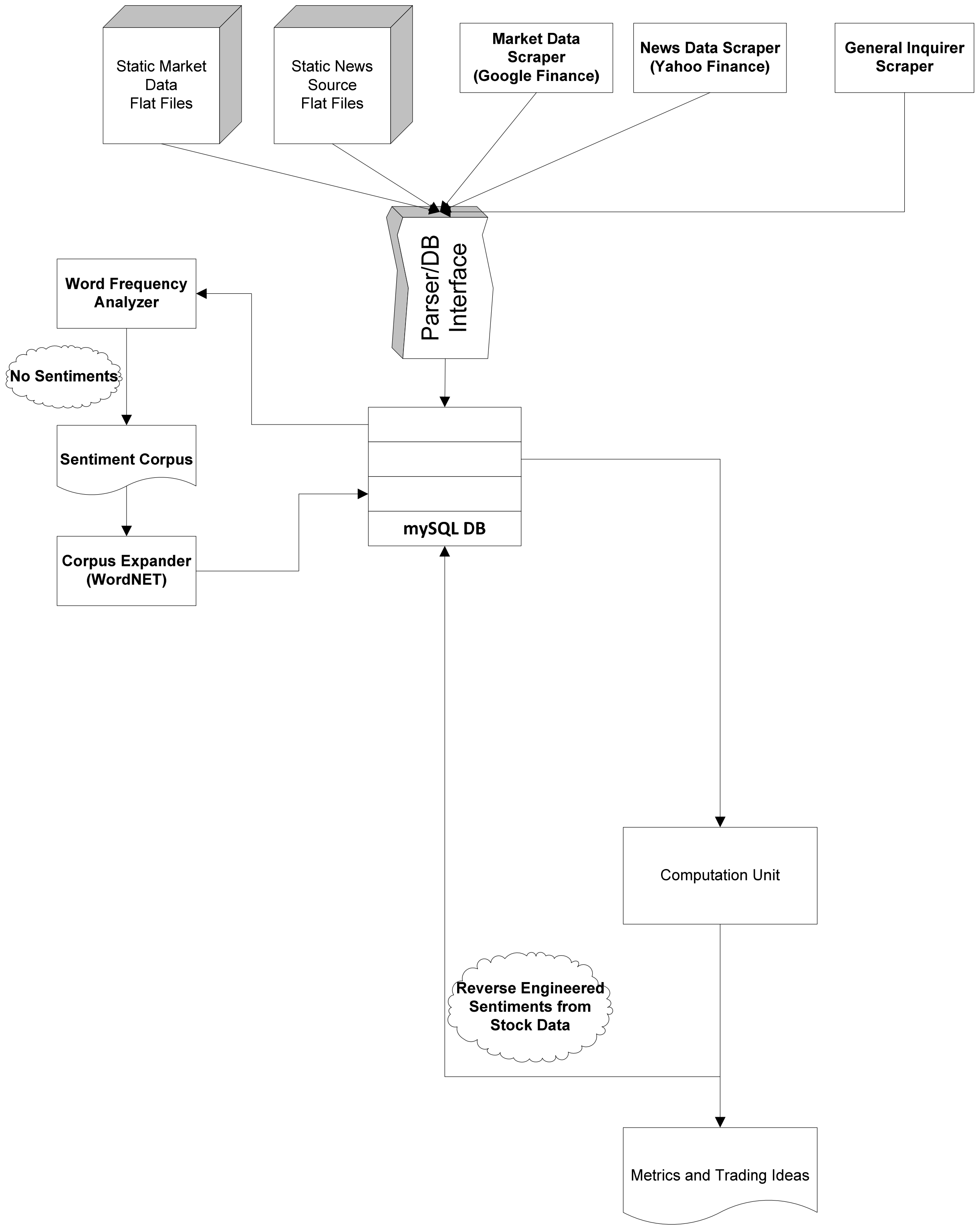
As we mentioned at the onset of the thesis, there are many directions in which NLP demonstrates much potential. One area we feel particularly strongly about is the idea of a crawler, as we alluded to in the *Related* Work section. A crawler would be an automated extension to our platform, which creates or extends sets of phrases which are highly related to one another. For example, suppose the parser is initialized with a sample set (“Citigroup”, “C”). The crawler could use websites such as TextMap in order to dynamically grow associations of the company “Citigroup” with individuals such as CEO “Charles Prince” after encountering that name in several news articles. We believe these *association buckets* could enhance the sentiment analysis discussed above, with the help of TextMap’s Sentiment monitor, which rates public *sentiment* (much like Billboard's Top 100 Hits) on high-profile individuals, such as CEO's of companies (Godbole, Srinivasaiah, & Skiena, 2007). We believe this idea has potential.

Appendix 1 – Phase 1

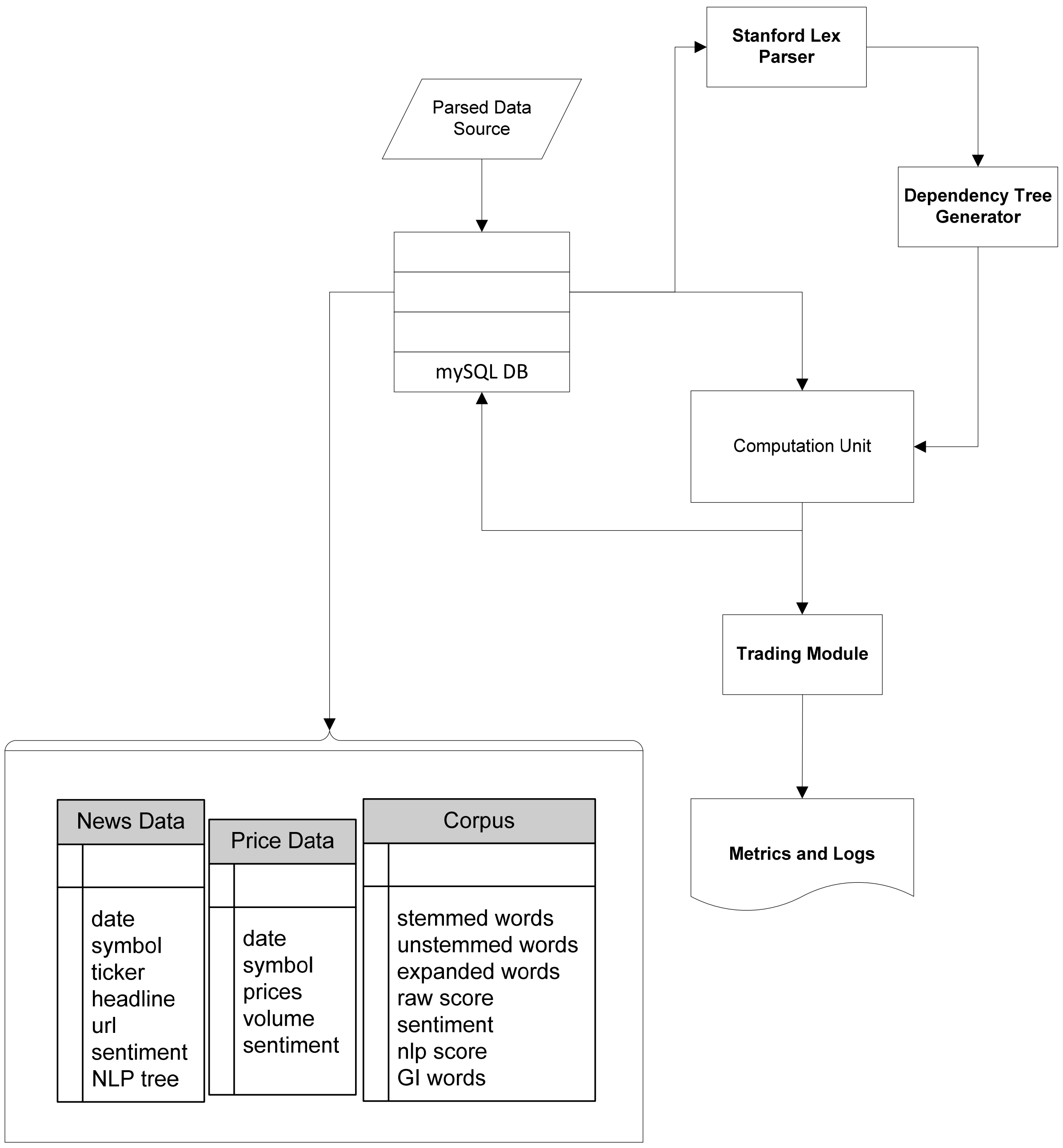
Bold represents the incremental changes made in that phase


Static News Source Flat Files
Parser Unit
Headline Frequencies
Static Market Data Flat Files
Computation Unit
Correlations & Performance Results
Metrics and Trading Ideas

Appendix 1 – Phase 2


Static Market Data Flat Files
Static News Source Flat Files
Market Data Scraper (Google Finance)
News Data Scraper (Yahoo Finance)
General Inquirer Scraper
Parser/DB Interface
Word Frequency Analyzer
No Sentiments
Sentiment Corpus
Corpus Expander (WordNET)
mySQL DB
Computation Unit
Reverse Engineered Sentiments from Stock Data
Metrics and Trading Ideas

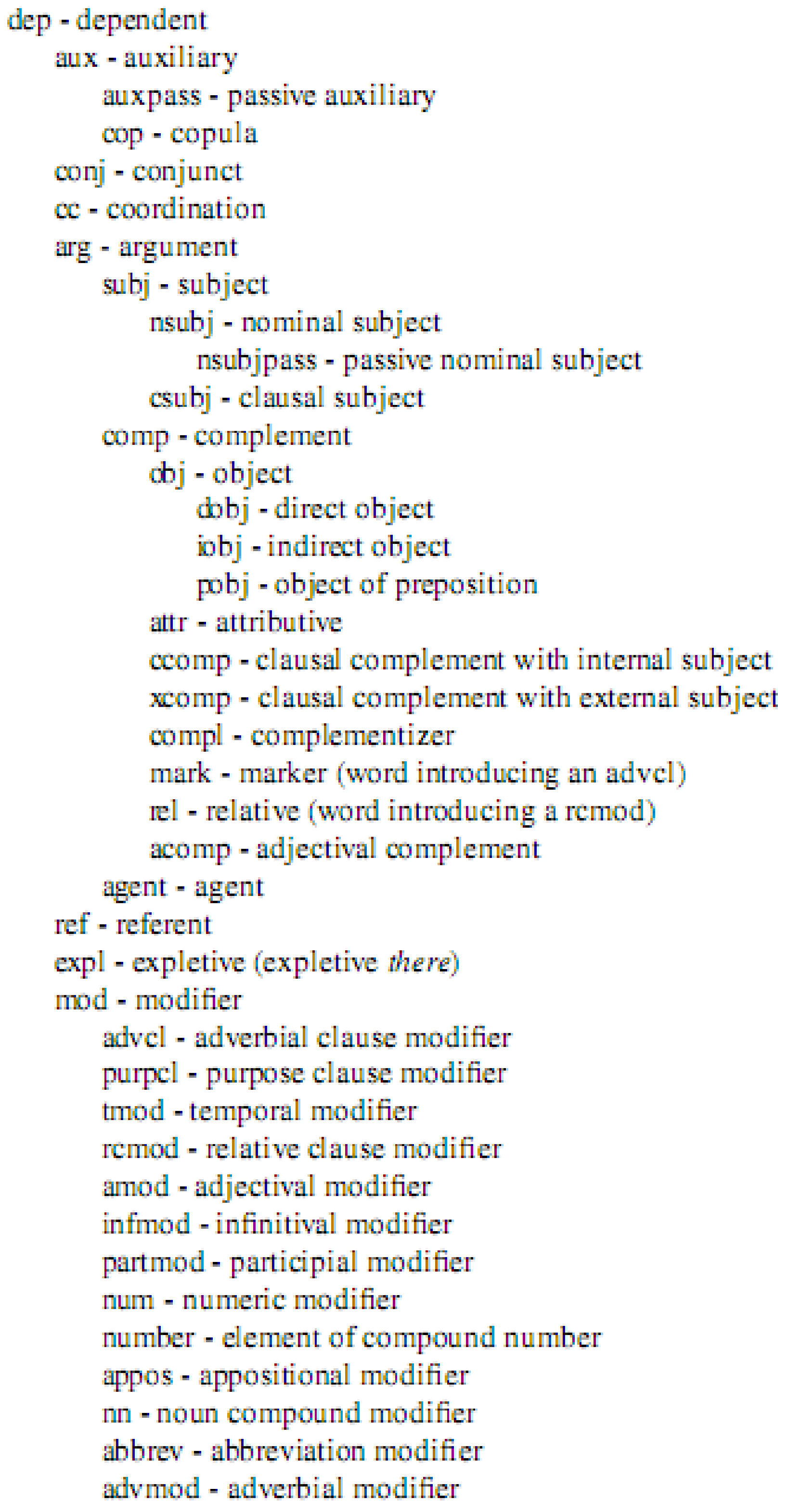
Appendix 1 – Phase 3



Appendix 2

	Smal Cap				Mid Cap				S&P 500			
	Ticker	Correlation of # Articles to Price Change	Ticker	Correlation of # Articles to Volume	Ticker	Correlation of # Articles to Price Change	Ticker	Correlation of # Articles to Volume	Ticker	Correlation of # Articles to Price Change	Ticker	Correlation of # Articles to Volume
Top 5	GHCI	0.6977	NWRE	0.0666	PNM	0.6243	NFLX	0.9383	KSE	0.6384	NVLS	0.8642
	EAGL	0.6875	CHAP	0.1626	BSG	0.5924	PPP	0.9195	TEX	0.5543	CPWR	0.7878
	EE	0.6686	INSP	-0.1172	WDC	0.5773	VCLK	0.8738	BC	0.5383	BOL	0.7517
	AH	0.6071	TWB	-0.3498	RPM	0.4920	ISRG	0.8455	MOT	0.5296	YUM	0.7312
	RHB	0.5974	CYMI	0.3625	LEA	0.4886	RDN	0.8205	AIZ	0.5228	CFC	0.7201
		....		....		....		....		....		....
Bottom 5	ANET	-0.5003	SFY	-0.2544	REG	-0.4329	EAC	-0.2242	BMC	-0.4485	WYN	-0.2700
	LG	-0.5089	KRG	-0.2547	SMG	-0.4404	HSIC	-0.2350	WM	-0.4741	BAC	-0.2965
	NNN	-0.5254	ALOG	-0.2828	WRI	-0.5069	AM	-0.2447	FO	-0.5552	MWV	-0.3301
	THQI	-0.5405	WGO	-0.3659	PWAV	-0.6206	CYTC	-0.2471	DJ	-0.6433	PRU	-0.3671
	CHG	-0.5602	SKT	-0.5035	PETM	-0.6708	AAP	-0.3119	CFC	-0.6621	AOC	-0.3680
	Average	*0.0084*		*0.2487*	*Average*	*0.0270*		*0.2685*	*Averaage*	*0.0619*		*0.2082*

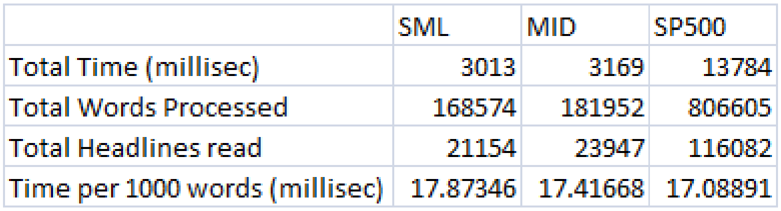
Appendix 3



- dep - dependent
  - aux - auxiliary
    - auxpass - passive auxiliary
    - cop - copula
  - conj - conjunct
  - cc - coordination
  - arg - argument
    - subj - subject
      - nsubj - nominal subject
        - nsubjpass - passive nominal subject
      - csubj - clausal subject
    - comp - complement
      - obj - object
        - dobj - direct object
        - iobj - indirect object
        - pobj - object of preposition
      - attr - attributive
      - ccomp - clausal complement with internal subject
      - xcomp - clausal complement with external subject
      - compl - complementizer
      - mark - marker (word introducing an advcl)
      - rel - relative (word introducing a rcmod)
      - acomp - adjectival complement
    - agent - agent
  - ref - referent
  - expl - expletive (expletive *there*)
  - mod - modifier
    - advcl - adverbial clause modifier
    - purpcl - purpose clause modifier
    - tmod - temporal modifier
    - rcmod - relative clause modifier
    - amod - adjectival modifier
    - infmod - infinitival modifier
    - partmod - participial modifier
    - num - numeric modifier
    - number - element of compound number
    - appos - appositional modifier
    - nn - noun compound modifier
    - abbrev - abbreviation modifier
    - advmod - adverbial modifier

Source: (Marneffe, MacCartney, & Manning, 2006)

Appendix 4

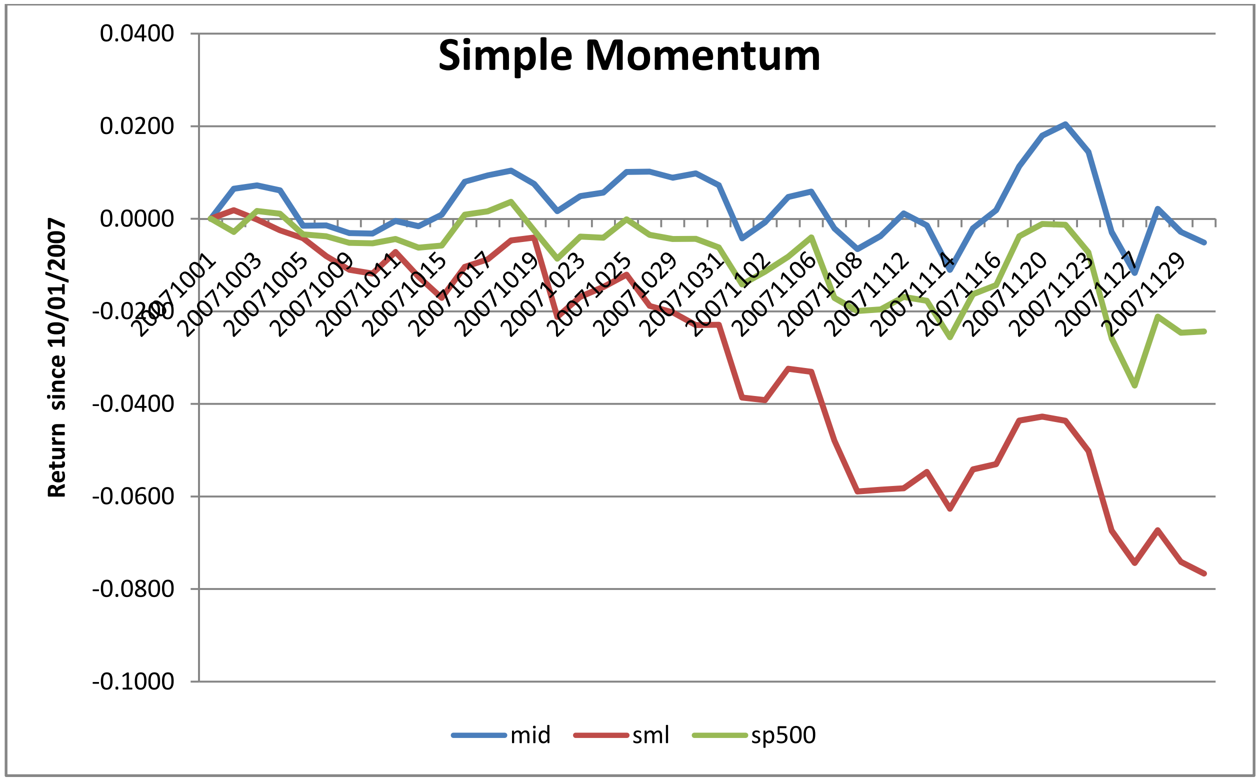


	SML	MID	SP500
Total Time (millisec)	3013	3169	13784
Total Words Processed	168574	181952	806605
Total Headlines read	21154	23947	116082
Time per 1000 words (millisec)	17.87346	17.41668	17.08891

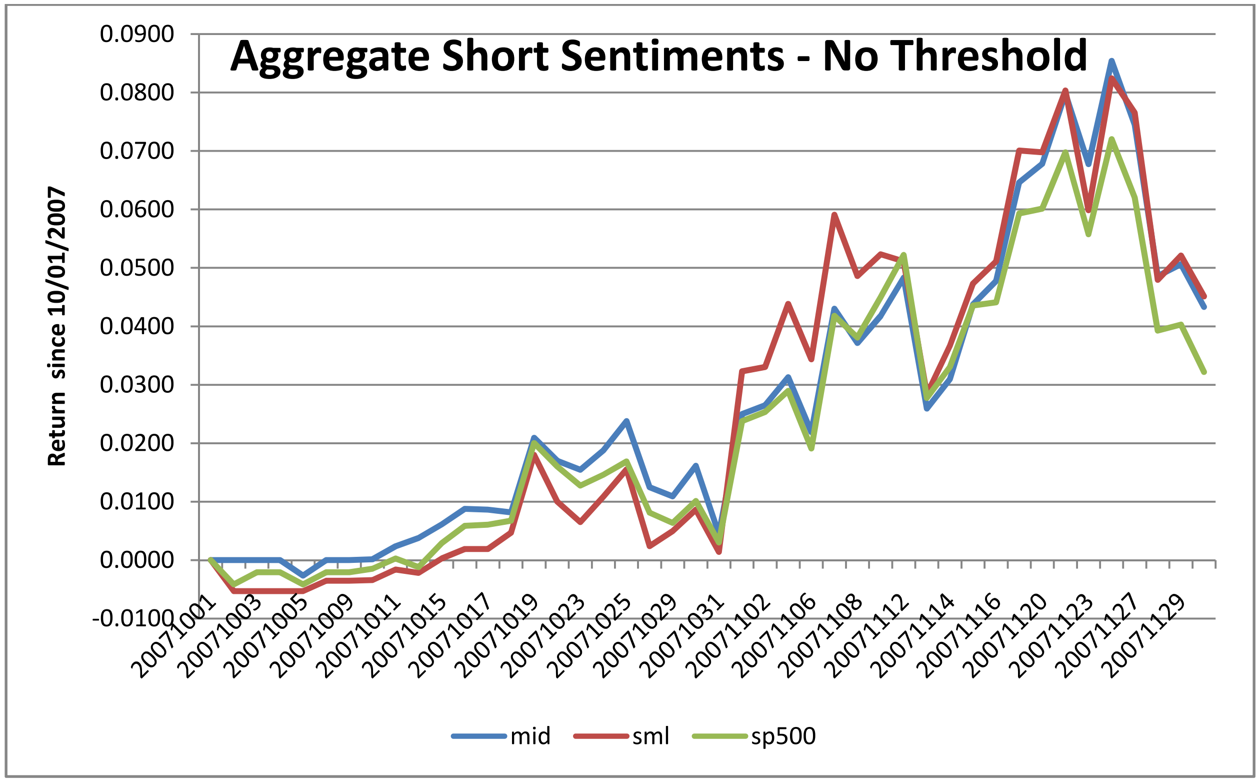
Appendix 5

Strategy Name	**Description of Decision Making (Opp. To *Sell* (*Hold* if not *Buy/Sell*))**
PLTS10 - Pure Momentum	Buy if stock price gained on previous day.
PLTS20 - Sent. Signal	Buy if sentiment of headline > 0.
PLTS21 - Sent. Signal Threshold	Buy if sentiment of headline > 1 stdev above historical mean sentiment.
PLTS22 - Sent. Expanded	Buy if expanded list of sentiment of headline > 0.
PLTS23 - Sent. Expanded Threshold	If expanded sentiment list > 1 stdev above historical mean sentiment.
PLTS24 - Sent. Respective Index	Buy if this sentiment > 0.
PLTS30 - Sent. Volume	If Δvolume from yesterday is 20% greater, follow rule PLTS20.
PLTS31 - Sent. NewsCount	ratio: sentiment/number of news events. If ratio < small threshold, buy.
PLTS40 - NLP: Sent.	Build dependency tree. If sentiment > 0, buy.
PLTS41 - NLP: G.I.	Build dependency tree with General Inquirer. If GI_sentiment > 0, buy.
PLTS42 - NLP: Sent. & G.I.	Build dependency tree with sentiment and G.I. If this sentiment > 0, buy.
PLTS50 - Market movement.	Overall market.
PLTS60 - Running Avg/StDev	Determine Threshold by keeping running averages/standard deviations.

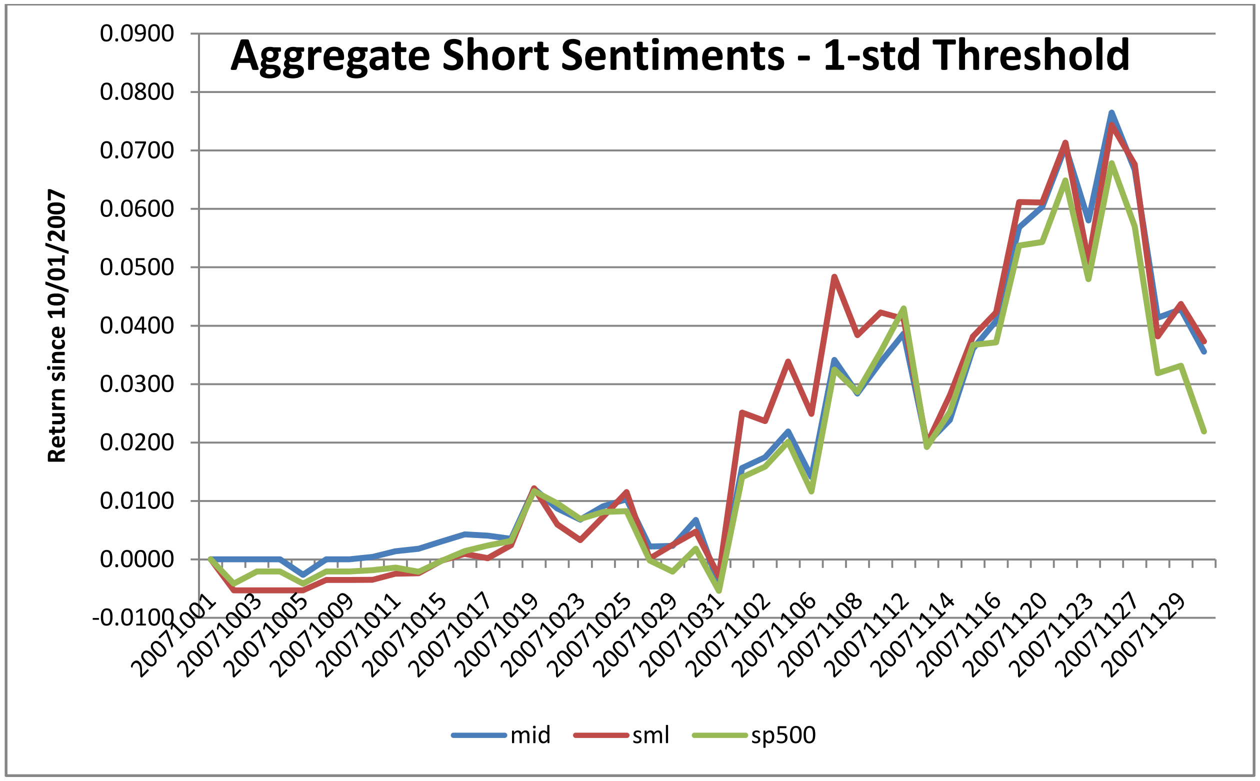
Appendix 6- *PLTS10*



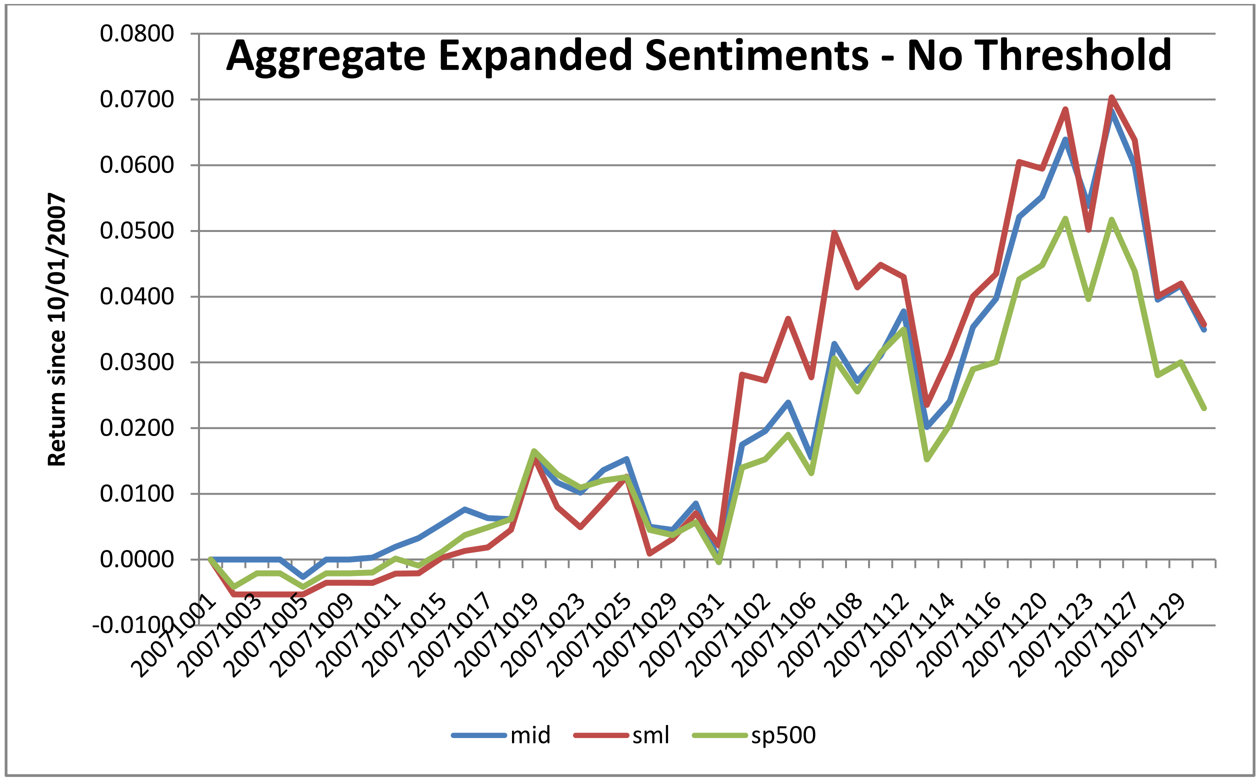
Appendix 6- *PLTS20*



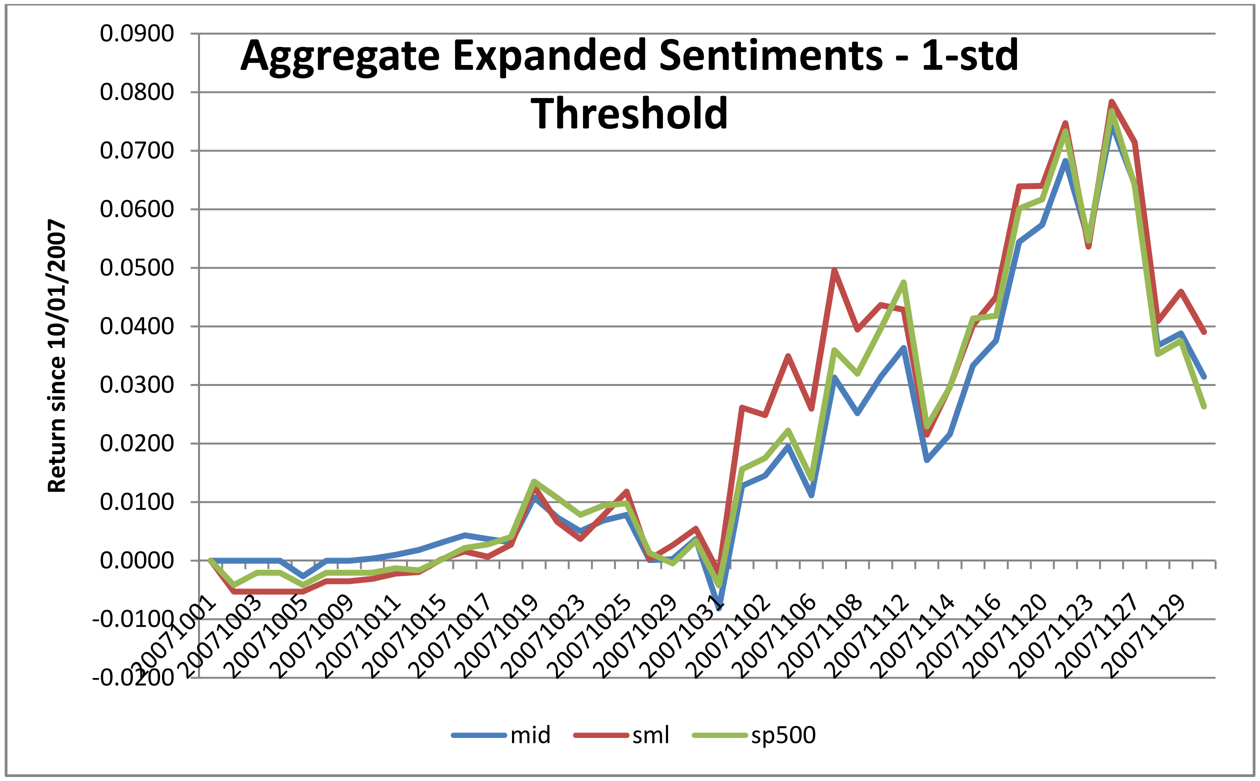
Appendix 6- *PLTS21*



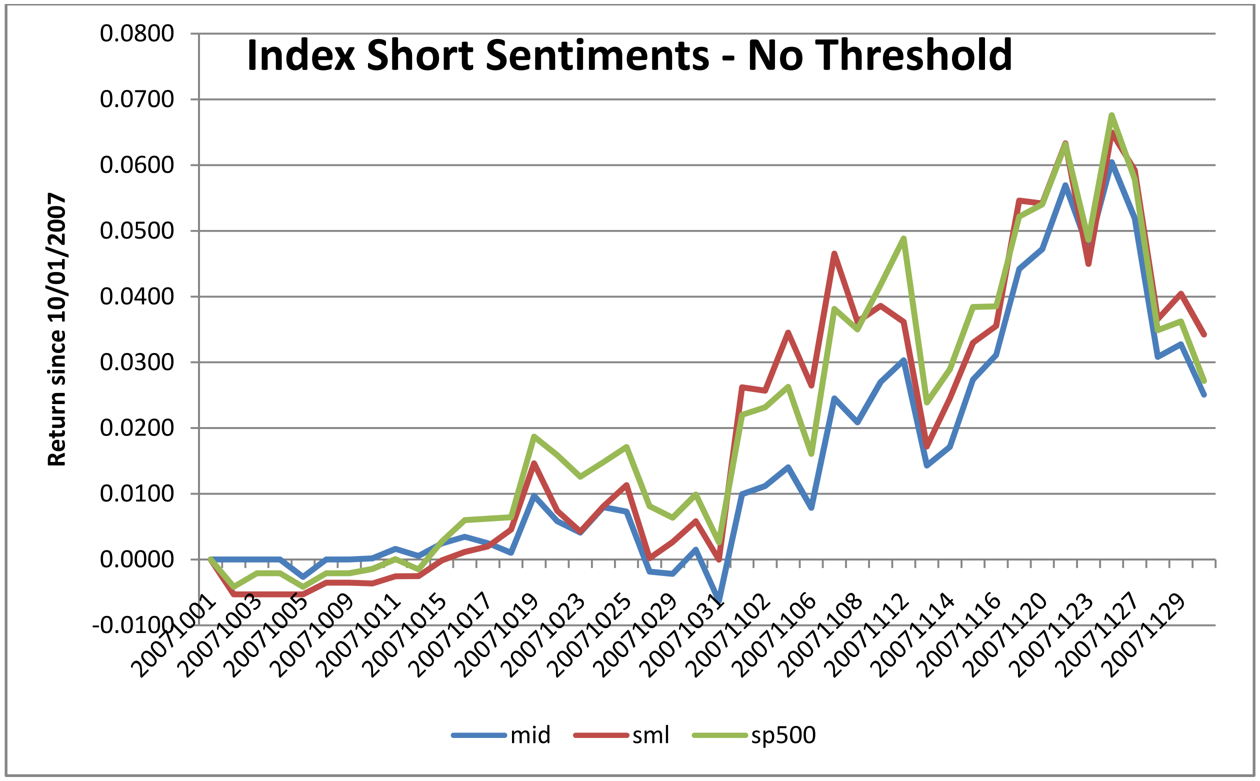
Appendix 6- *PLTS22*



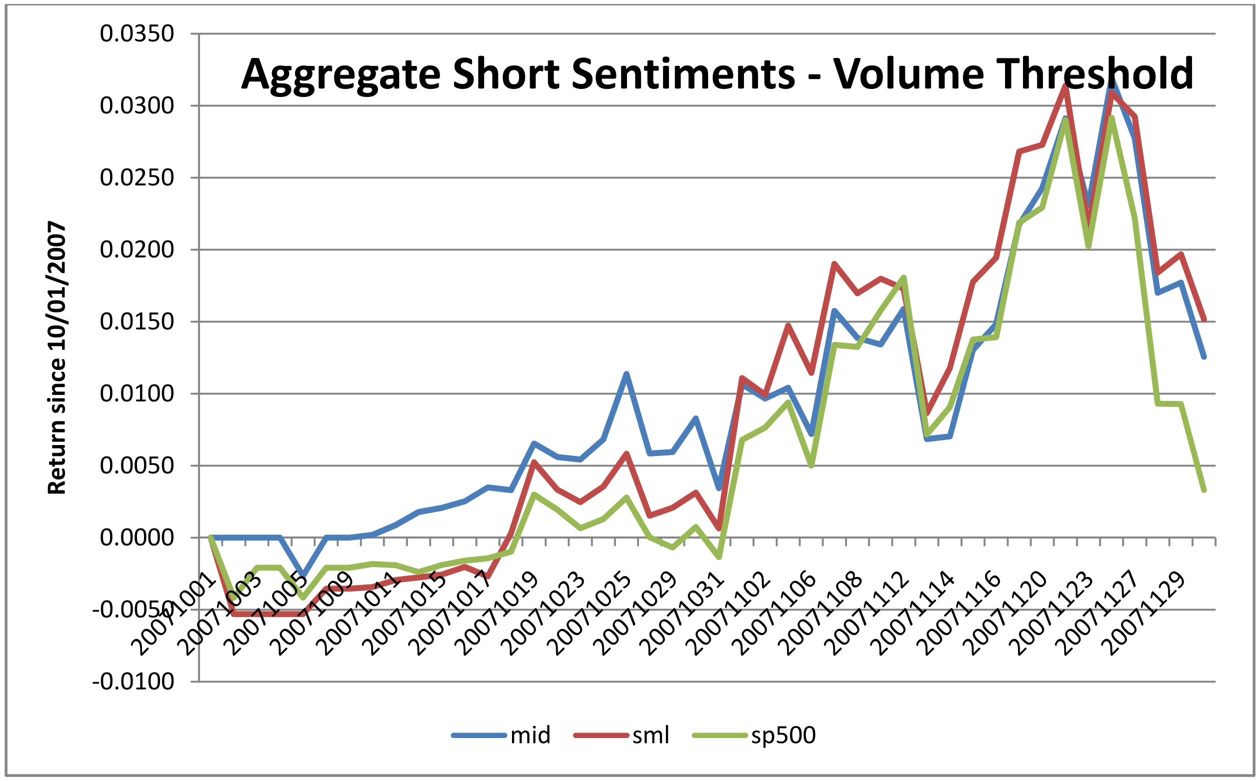
Appendix 6- *PLTS23*



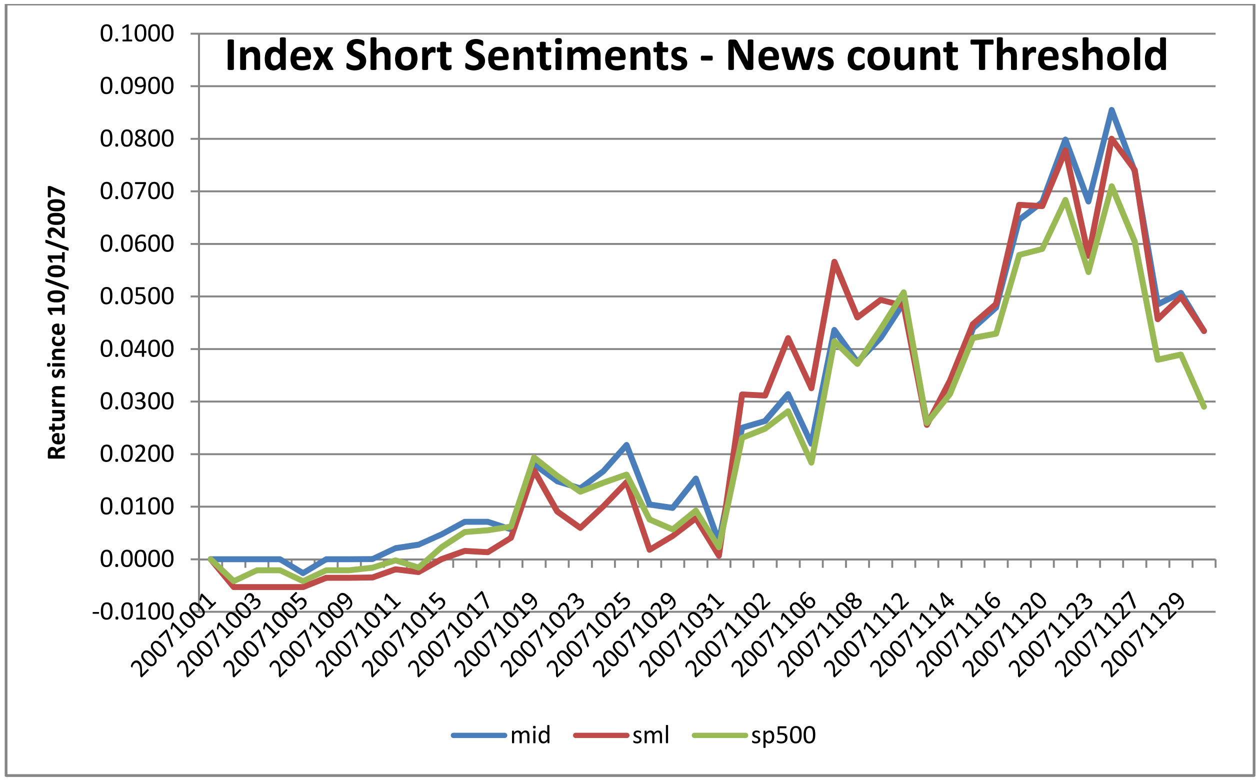
Appendix 6- *PLTS24*



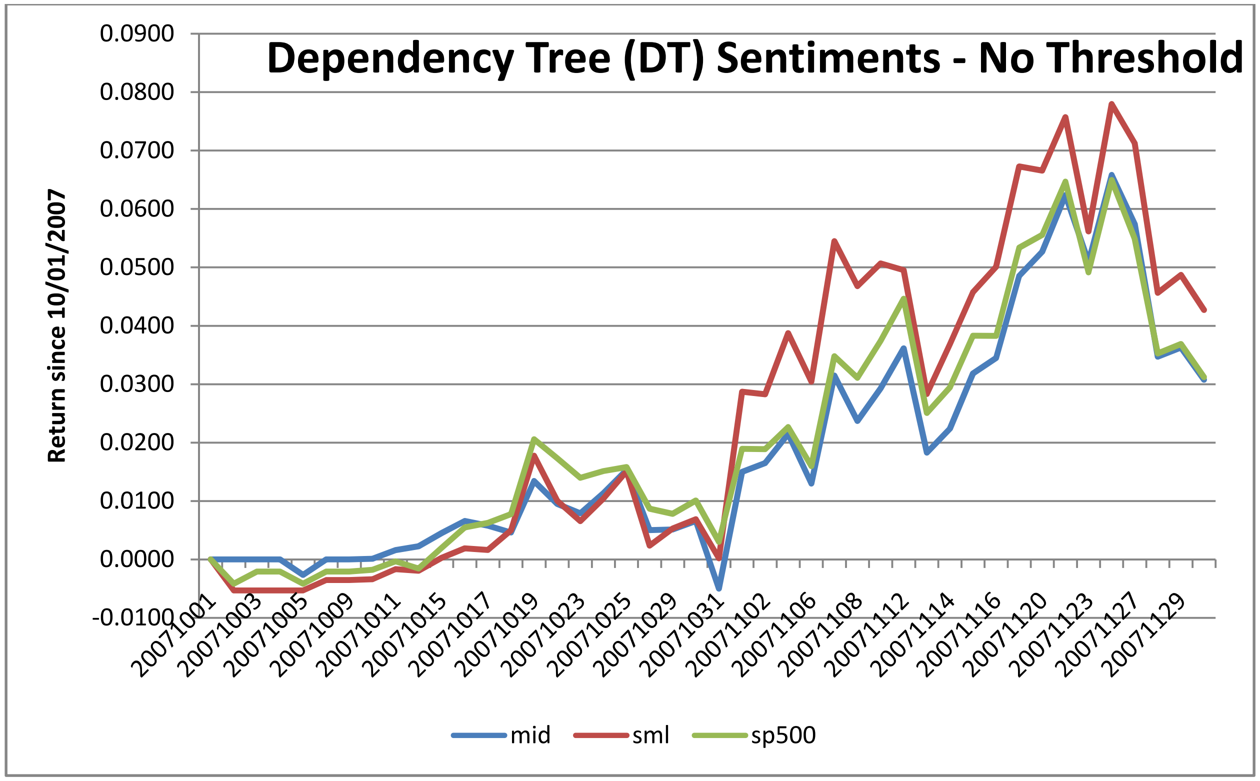
Appendix 6- *PLTS30*



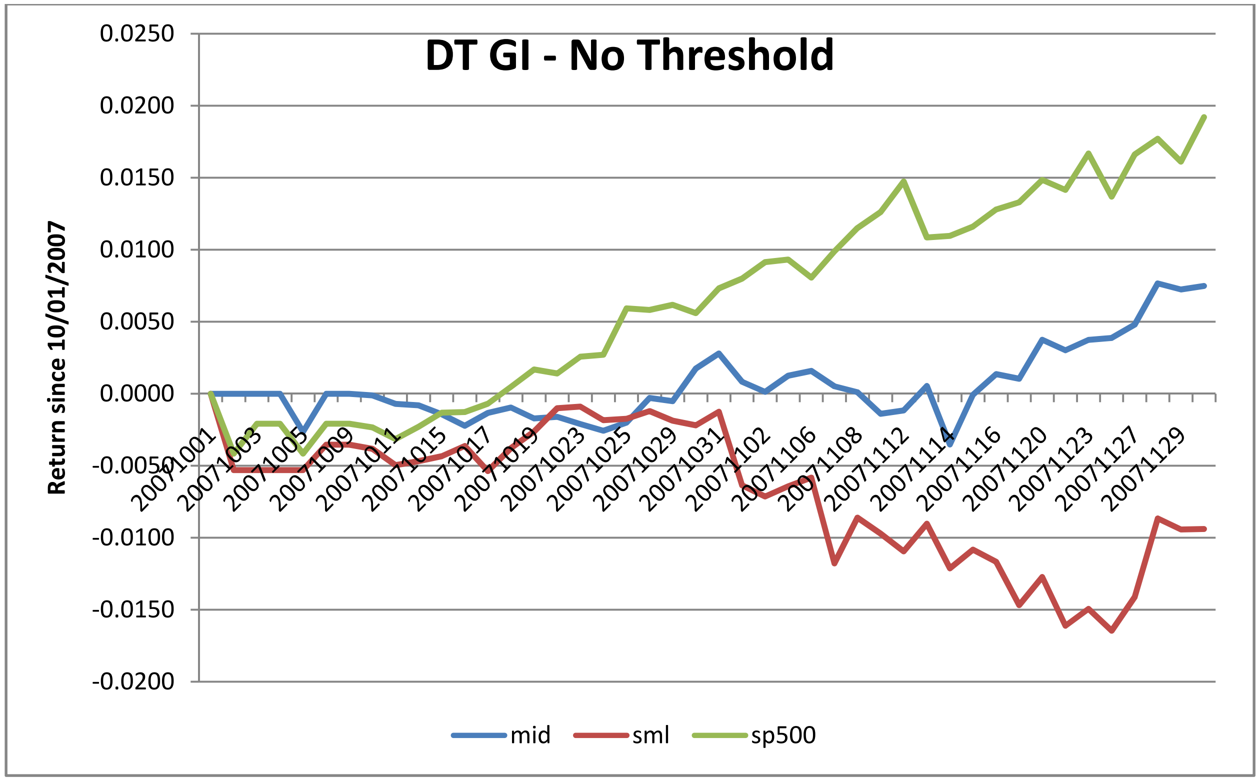
Appendix 6- *PLTS31*



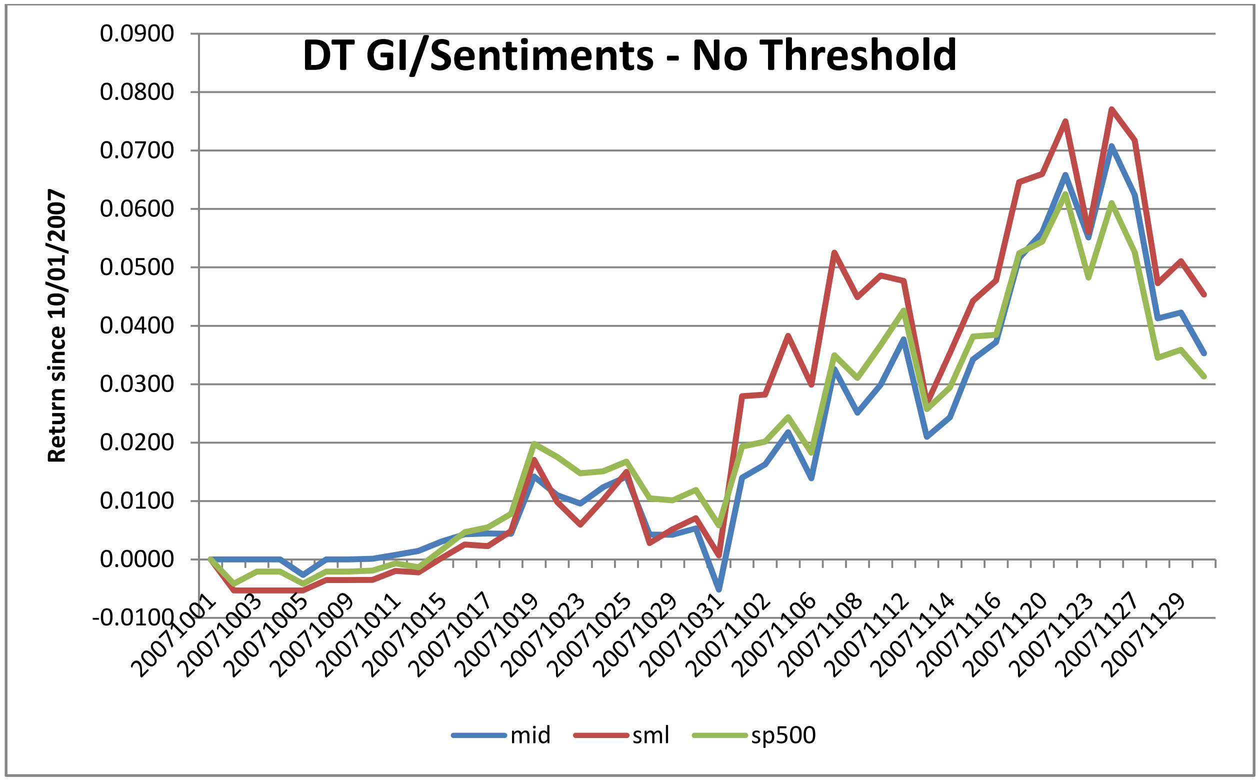
Appendix 6- *PLTS40*



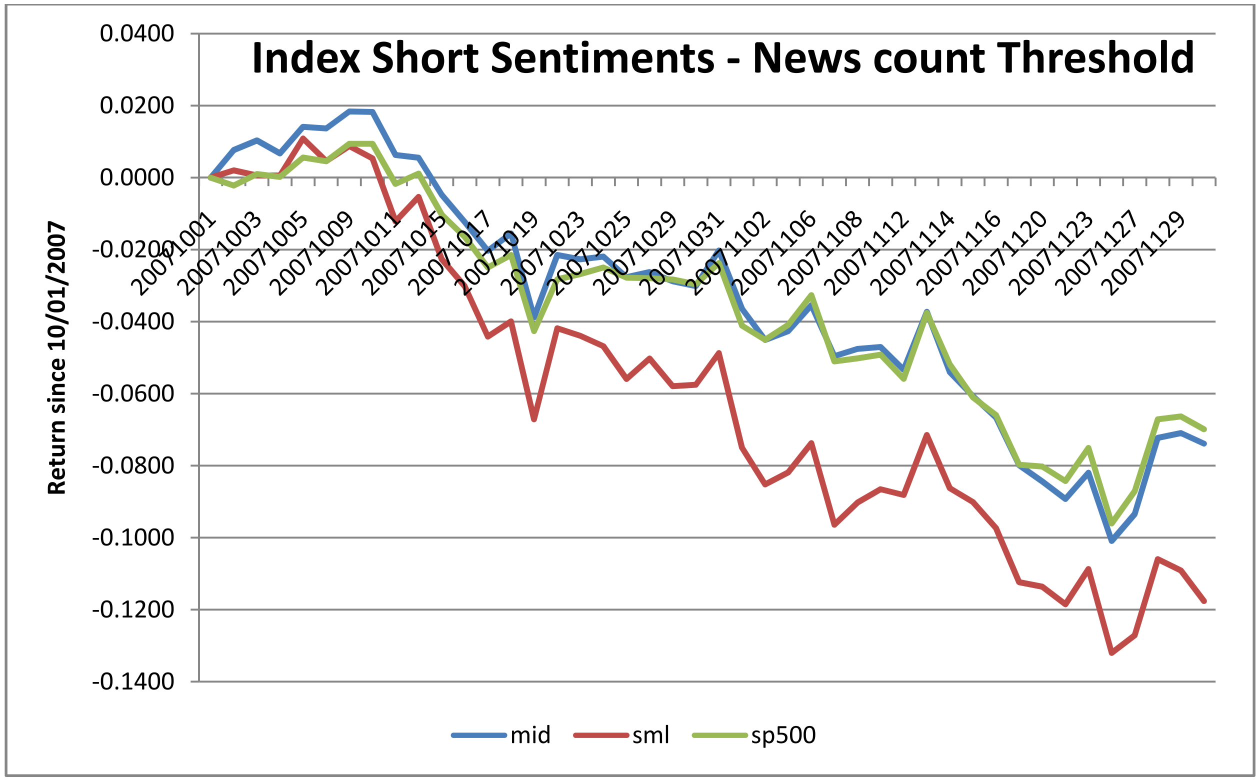
Appendix 6- *PLTS41*



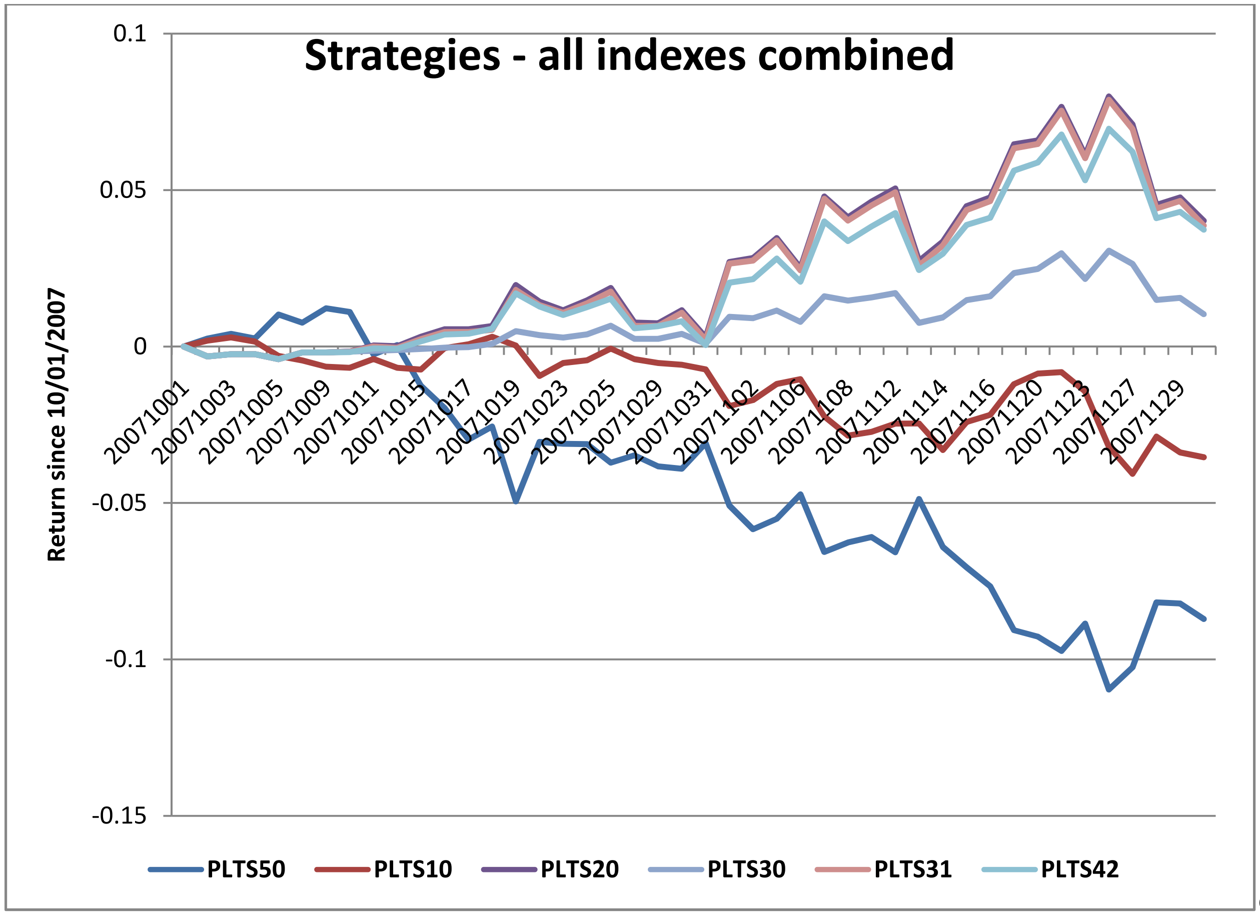
Appendix 6- *PLTS42*



Appendix 6- *PLTS50*



Appendix 6- *Strategies aggregated over three indexes*



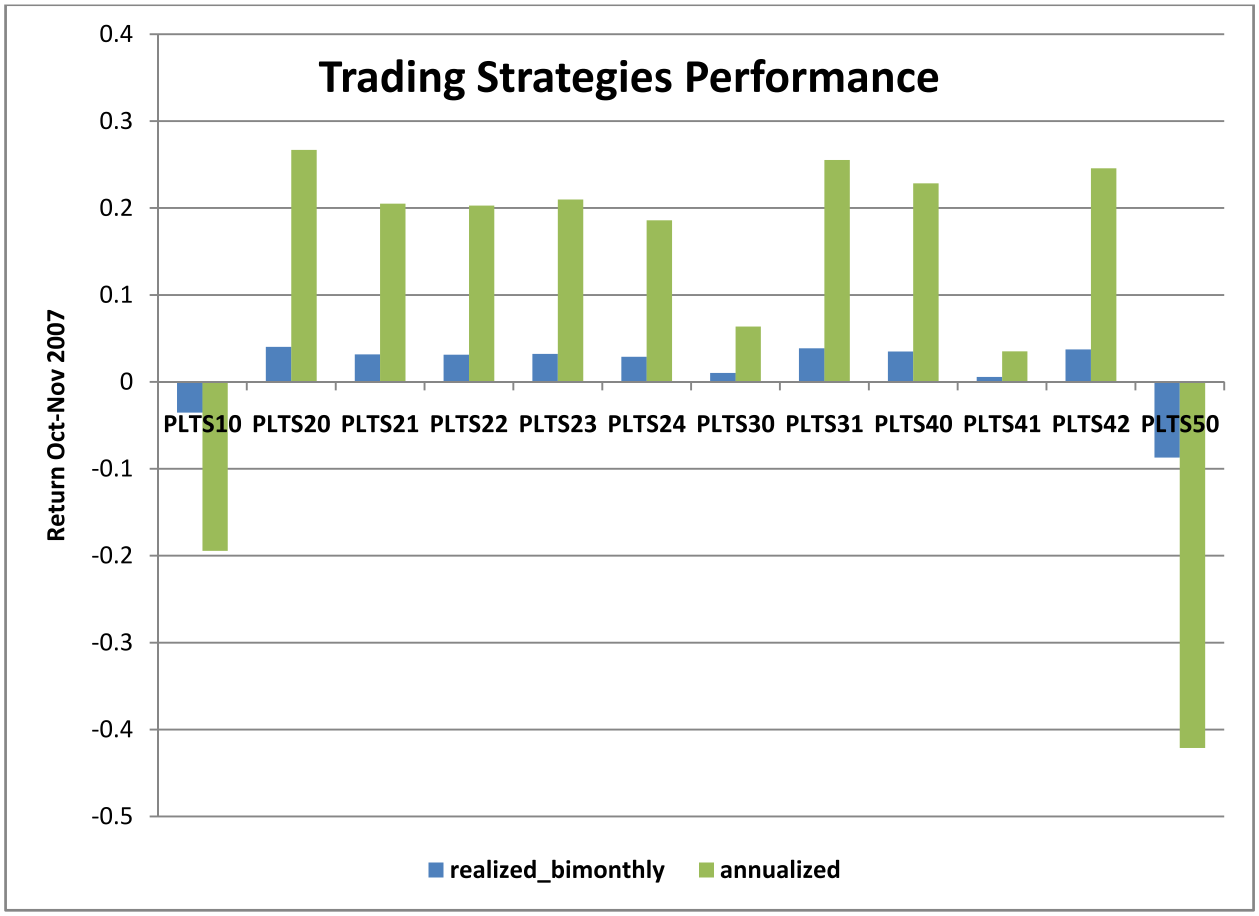
e

Notes: This graph represents returns for the four strategies indicated against the benchmark market averaged across the three indices. The individual graphs for each index look nearly identical.

Appendix 7

Return	PLTS10	PLTS20	PLTS21	PLTS22	PLTS23	PLTS24	PLTS30	PLTS31	PLTS40	PLTS41	PLTS42	PLTS50
2-Mo.	-3.5%	4.0%	3.2%	3.1%	3.2%	2.9%	1.0%	3.9%	3.5%	0.6%	3.7%	-8.7%
APY	-19.4%	26.7%	20.5%	20.3%	21.0%	18.6%	6.4%	25.5%	22.9%	3.5%	24.6%	-42.1%

Note: The tables compares the 2 month and annualized return (averaged over the three indexes) for the different strategies. PLTS50 is the overall market, which had a return of 8%.



Note : This graph represents returns for strategies averaged across the three indices.

References

Buvac, V. (2002, September 12). General Inquirer. Retrieved April 4, 2008, from General Inquirer: http://www.webuse.umd.edu:9090/

General Inquirer is a NLP tool that helps tag words into different categories.

Fallows, J. (2004, December 26). At I.B.M., That Google Thing Is So Yesterday. The New York Times. <http://www.nytimes.com/2004/12/26/business/yourmoney/26techno.html?ex=1261717200&en=506a267f2a622db5&ei=5090> .

The journalist describes advancements made in machine-abstraction, particularly via NLP, at some of the most salient technological institutions today. The article describes a model in which an entire passage is studied in order to abstract meaning, or *the gist,* of the paragraphs. Of particular importance is the fact that the specific words of interest need not appear in the actual passage. So long as the machine has enough associations with the central theme, it is capable of *reading between the lines*. In computer science terms, this is simply increased accuracy of word associations; however, the potential applications in the human world are what make the bulk of the article. While slightly dated, in terms of the typical *age* of research we encountered, it is nonetheless a high-level overview of the applications of NLP. In fact, this was one of the earlier sources during our research, which truly helped us fathom an application of NLP into the realm of financial markets.

Gilder, A., & Lerman, K. (2007, May). Reading the Markets: Forecasting Prediction Markets By News Content Analysis. University of Pennsylvania.

The study focused on application of time-series and context based analysis of market data and news releases, respectively, in order to select an optimal trading strategy. The scope of these markets included non-financial assets such as, bets on results of an election, in order to gauge accuracy of the predictive model. This paper shed light on the degree of NLP we should aim to achieve - as their results indicated the performance of a naïve bag-of-words analysis and a more sophisticated context-based approach. The success of the part-of-speech parser, as conjectured by the authors and demonstrated in the study, has given us at the very minimum, a method against which to check our benchmark. The authors were very recently pursuing a senior design thesis, and the originality of their idea and implementation met the approval of staff at University of Pennsylvania's CIS department.

Gillam, L., Ahmad, K., & Ahmad, S. (2002). Economic News and Stock Market Correlation: A Study of the UK Market. <http://www.saifahmad.com/tke.pdf> . U.K.: University of Surrey, Department of Computing.

The authors, professors at the University of Surrey, present a study of news article correlation to the UK markets. The article is bit older and we are not sure how much of their study still applies to the US markets today. However, they are worthy of noting and can potentially help us. They used a set of 70 words and found how often they occurred in news articles. They only looked at the index and news article on a broader market; this is one area of difference to our approach. Their approach, though not as deep as others, does provide preliminary results, similar to our phase 1. Their results were not as good as someone manually reading the article and they argue more work is need. This is similar to our approach and the motivation for phase 2 and 3.

Godbole, N., Srinivasaiah, M., & Skiena, S. (2007). Large-Scale Sentiment Analysis for News and Blogs. International Conference on Weblogs and Social Media. Boulder.

The paper written by members of the Google team as well as a professor from SUNY provide methods for developing a set of sentiment words/phrases as well, the paper was written last year. They built the set of sentiment words by using path. Starting with some core words, they looked at the synonyms and antonyms and created paths from one word to another. The longer it takes to get to a different word, the less their sentiments are similar. Once they had sentiments, their significance was found using the score each sentiment receives in the graph following some rules they set out. They also used co-occurrence of sentiments and words in same sentence to evaluate the importance of different words. The authors claim their successful results are based on large set of text they processed. They say that news and blogs data about stock markets can be shown to be correlated to their evaluation of sentiments. Their results might helps us implement a similar approach, however they claim that resources required to process vast amounts of text is significant so that might be a constraint on us.

Greenwald, A., Jennings, N. R., & Stone, P. (2003). Agents and Markets. IEEE Intelligent Systems. <http://ieeexplore.ieee.org/iel5/5254/27968/01249164.pdf> , 12-14.

The authors, from Brown University, University of Southampton, and University of Texas, point out the advantages of having an agent (automated trading) versus a human trader. They present multiple reasons for agents, including more calculations, not distracted, and can make them immune to flaws. They also argue that though automatic agents can process more data, more data is not always a good thing. Given this, we realize that our system can process lot of data, however in the real market others might have similar machine so we need to be aware of such issues as cooperation and competition. Essentially, it could be that many other systems do a similar thing and because they might be faster, there are no real opportunities left in the market to act on.

Group, S. N. (n.d.). The Stanford Natural Language Processing Group . Retrieved April 3, 2008, from Stanford University: http://nlp.stanford.edu/

The Stanford NLP package has many advanced features that can be used with NLP analysis. We primarily used the parser which can be used to build dependency trees.

Hariharan, G. (2004, May). News Mining Agent for Automated Stock Trading. Google Cache <http://www.ece.utexas.edu/~guru/news_mining_agent.pdf> . University of Texas at Austin.

The author wrote the article in 2004 when he was a student at University of Texas. He discusses automated trading how news stories can be used to predict market behavior. His discussion of how information can be used to trade predicatively is similar to our project. In the last step of our project, the author's discussion and the modification he made to his system was useful. However because the article was written in 2004 it is a bit dated and we are not how the modification would still hold up. Once we reach phase 3, we will then have a better assessment of how well his suggestion hold up.

Kaji, N., & Kitsuregawa, M. (2007). Building Lexicon for Sentiment Analysis from Massive Collection of HTML Documents. Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, (pp. 1075-1083). Prague.

The paper is presented by professors from University of Tokyo who have an interest in data mining. They argue that sentiment of words can be determined using two methods, thesaurus which uses synonymous to determine the polarity of word. However, the professors argue the second method, which they present in the paper, is to determine polarity on structure and co-occurrences of words. Using some 1 billion HTML pages they were able to achieve a success rate of 92% in deciding the polarity of word and phrases by just using the structure of the sentence and co-occurrences. This approach is stronger than other because allow for new word's polarity to be determined from structure. Given the success rate their achieved it and if we can imitate it in our project it would lead to some significant results, however their underlying data source was Japanese HTML pages so we are not sure how well it will work with short headlines that can be ambiguous.

Kelly, J. (2007, June). The Ultimate Money Machine. Bloomberg Markets. <http://www.cis.upenn.edu/~mkearns/umm.pdf> , pp. 36-48.

The article, appearing in a leading Wall Street publication, highlights what the last decade has seen in terms of increasing influence of modern computing techniques in shaping the financial world. At the heart of the article is the growing power (intelligence and performance) of computing. As processor speed has enabled increased computation, quantitative minds as described in the article, have invented innovative applications of mathematical and linguistics analysis of financial markets. Professors at leading universities are helping financial firms develop these models, which employs their research in the field of Computer Science. While minimal in terms of technical implementation, the information in this article has had a two-fold impact on our project. First, it immediately focused our attention on applications of language processing in the context of finance, and secondly it inspired us to approach Professor Michael Kearns, our Faculty Advisor.

Kim, S.-M., & Hovy, E. (2004). Determining the Sentiment of Opinions. International Conference on Computational Linguistics. Geneva: Association for Computational Linguistics.

The professors from University of Southern California present on detecting sentiment from a sentence. Their approach it to take the sentence to be analyzed, breaks it down into its components, noun, verb...etc, and then build the sentiment from that. They initially build the set of sentiments manually and then grow it using WordNet. This approach seems simple enough and we did something similar given the resources we had. They did end with some good result; however the professors end the paper saying more work is needed; however their approach is a start on the right path.

Koppel, M., & Shtrimberg, I. (2006). Good News or Bad News? Let the Market Decide.

Koppel and Shtrimberg present their analysis on labeling articles regarding publicly traded companies as positive or negative. They argued that by looking at the immediate price in the stock, the news can be labeled positive or negative with an accuracy of about 70%. We found this article helpful as they presented methods for detecting positive or negative change which we were able to implement in our sentiment analysis.

Marneffe, M.-C. d., MacCartney, B., & Manning, C. D. (2006). Generating Typed Dependency Parses from Phrase Structure Parses.

The paper describes the system for extracting type dependency relationships from phrases and sentences. This paper complements the Stanford Lex Parser. It also gives an overview of the 48 relationships that are an output of the Stanford Lex Parser.

Meena, A., & Prabhakar, T. V. (2007). Sentence Level Sentiment Analysis in the Presence of Conjuncts Using Linguistic Analysis.

Meena and Prabhakar make use of General Inquirer and Wordnet to extract sentiments of words in a given phrase or a sentence. Their main focus is on the use of conjuncts as rooted pointed. They developed rules regarding different conjunctions and apply them to their dependency trees to come up with a sentiment score. Overall they have had success of 80%. The authors' work has been referenced in many places and we found this article to be useful reliable in our analysis.

Miller., G. A. (n.d.). WordNet. Retrieved April 4, 2008, from Princeton University: http://wordnet.princeton.edu/

WordNet is a software package that provides functionality for NLP. We used it primarily to retrieve the stem of a word as well as its synonyms.

Skiena, S. (2007). TextMap provided by The Research Foundation of State University of New York. Retrieved September 20, 2007, from TextMap: <http://www.textmap.com>

The author and co-developer of the website is a professor at SUNY, heavily involved in Computational Linguistics. The TextMap website can be best thought of as a toolbox, providing the capability of querying different metrics on various entities based predominantly on blog sentiment analysis. One such tool of particular interest to our study is its sentiment analysis tool: it enables the user to detect how the general population *feels* towards, say, Hillary Clinton, today. By tracking this sentiment analysis as a time series for various companies, their CEO's, or their general product, it is possible to more accurately assign keywords, a sentiment charge. The website publicly made available its data, fairly recently, and the founder is also an active contributor to the field of NLP (in fact, he is the co-author of another paper we reference). Due to the complexity of the initial NLP analysis (context based), we are uncertain whether we will be able to use TextMap as part of our prototype; however, it is interesting regardless.

Subramanian, H. K. (2004, December). Evolutionary Algorithms in Optimization of Technical Rules for Automated Stock Trading. <http://www.cs.utexas.edu/~pstone/Courses/395Tfall06/resources/Thesis-harish.pdf> . University of Texas at Austin.

The author was a Master's candidate at the University of Texas, and he presents this as a Master's thesis. He argues that technical approaches to stock trading can lead to losses. Therefore he suggests using a mix of technical strategies. The work of author is of interest to us because his methods of modifying technical strategies will help us fine tuning our third phase. However, his paper is slightly dated and we are not sure how relevant it might still be. Unlike other research papers we have looked at, none of them have discussed modifications to the technical approach; therefore, this article might thus come in handy.